A comparison of two 32bx32b signed multipliers. “Baugh-Wooley” and “pre-sum before Dadda tree using Baugh-Wooley”.

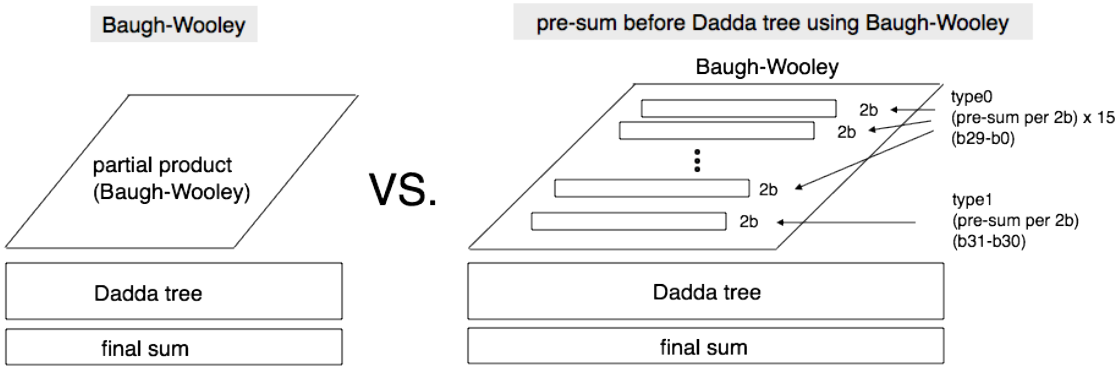
Hiro Mori
Dec.29,2023
gmail:bqe10133@gmail.com

SUMMARY

“pre-sum before Dadda tree using Baugh-Wooley” is 18% smaller than “Baugh-Wooley”. The delays are the same. The results are similar to the results of unsigned 32bx32b multiplier.[3]

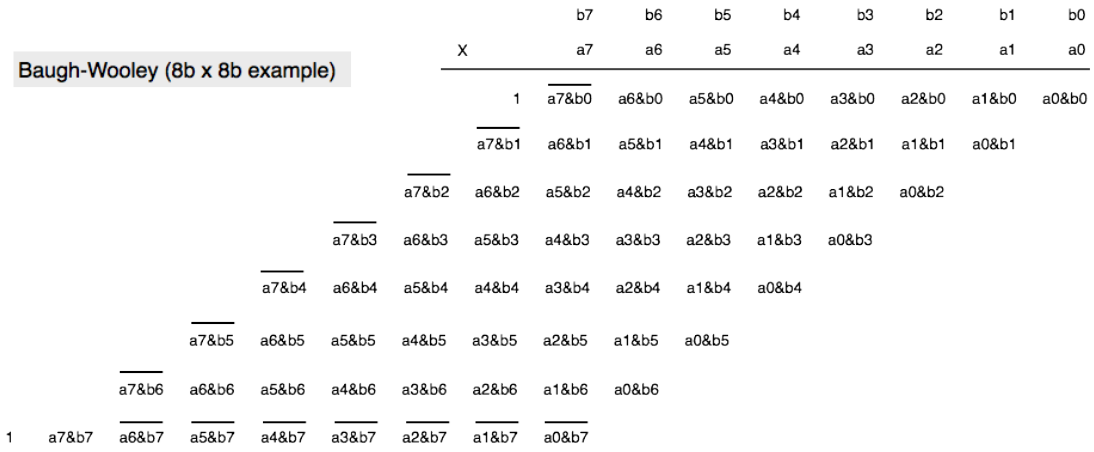
gate count		
Baugh-Wooley	pre-sum before Dadda tree using Baugh-Wooley	percentage %
11,166	9,074	81.3%

gate delay		
Baugh-Wooley	pre-sum before Dadda tree using Baugh-Wooley	percentage %
58	58	100.0%

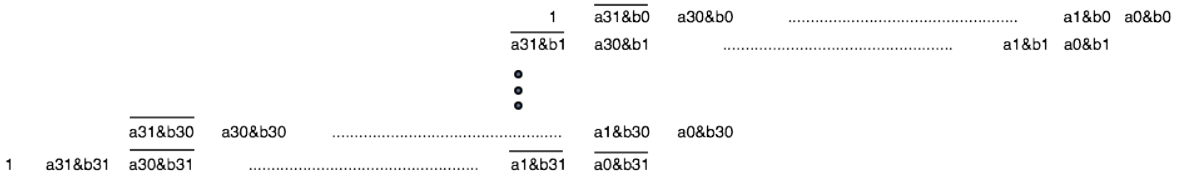


1. “Baugh-Wooley” multiplier

1.1 multiplier diagram

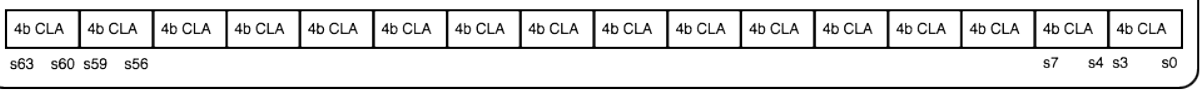


partial product 32 x 32 + 2 = 1026

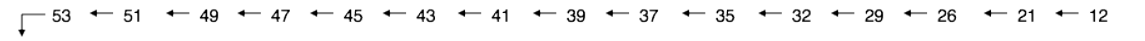


Dadda tree 32 -> 22 -> 15 -> 10 -> 7 -> 5 -> 4 -> 3 -> 2 900 full adder, 30 half adder

final sum 16 x (4b CLA)

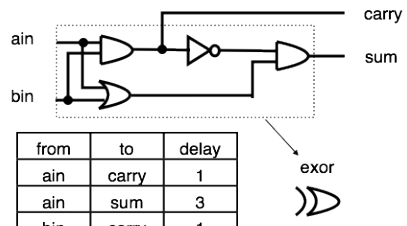


carry delay



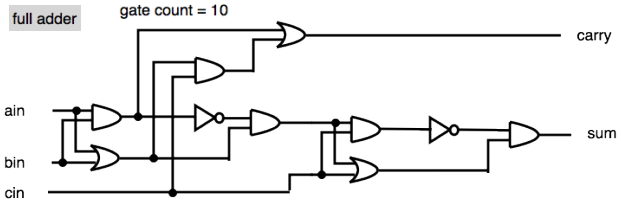
worst delay = 58 (s63, s62, s61)

half adder gate count = 4



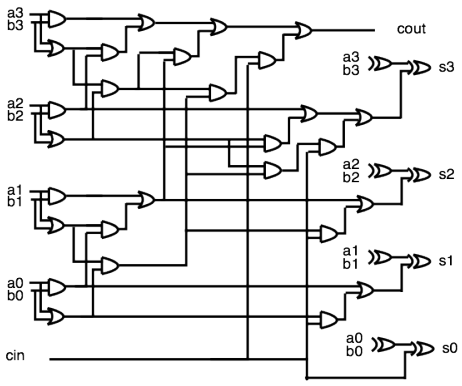
from	to	delay
ain	carry	1
ain	sum	3
bin	carry	1
bin	sum	3

full adder gate count = 10



from	to	delay
ain	carry	3
ain	sum	6
bin	carry	3
bin	sum	6
cin	carry	2
cin	sum	3

4bit CLA gate count = 60



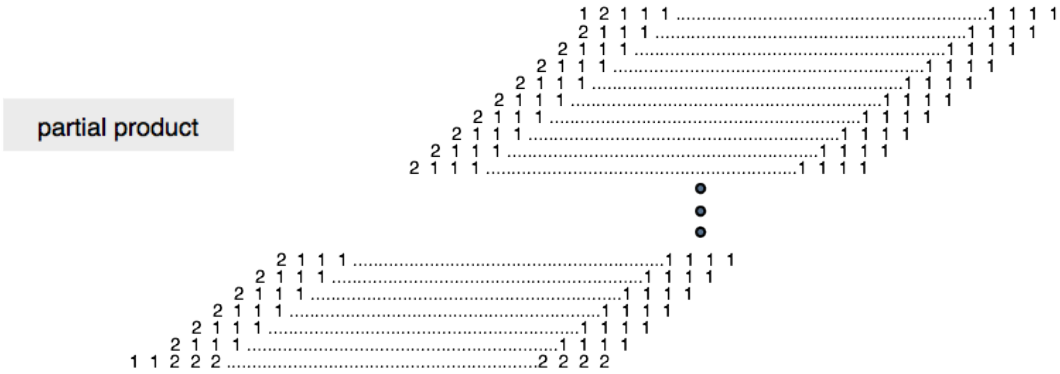
	cout	s3	s2	s1	s0
a3,b3	5	6	-	-	-
a2,b2	5	7	6	-	-
a1,b1	6	9	7	6	-
a0,b0	6	9	7	6	6
cin	2	5	5	5	3

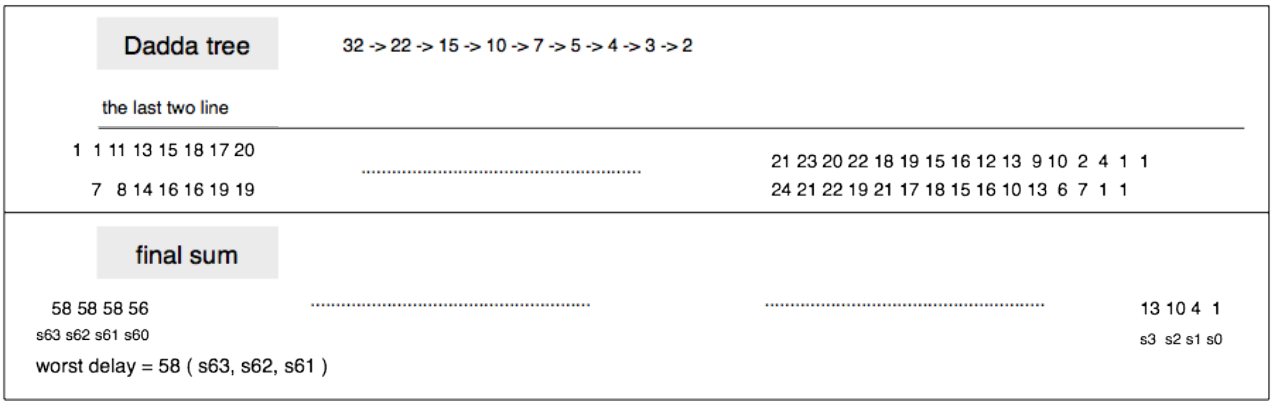
1.2 gate count

```
[1] partial product
(AND gate(= 1 gate count) x 32 x 32) + (31 x 2)(=negation of AND) = 1,086

[2] Dadda tree
full adder(= 10 gate count) x 900 = 9,000
half adder(= 4 gate count) x 30 = 120

[3] final sum
4bCLA adder(= 60 gate count) x 16 = 960

[4] total gate sount
1,086 + 9,000 + 120 + 960 = 11,166
```

1.3 gate delay

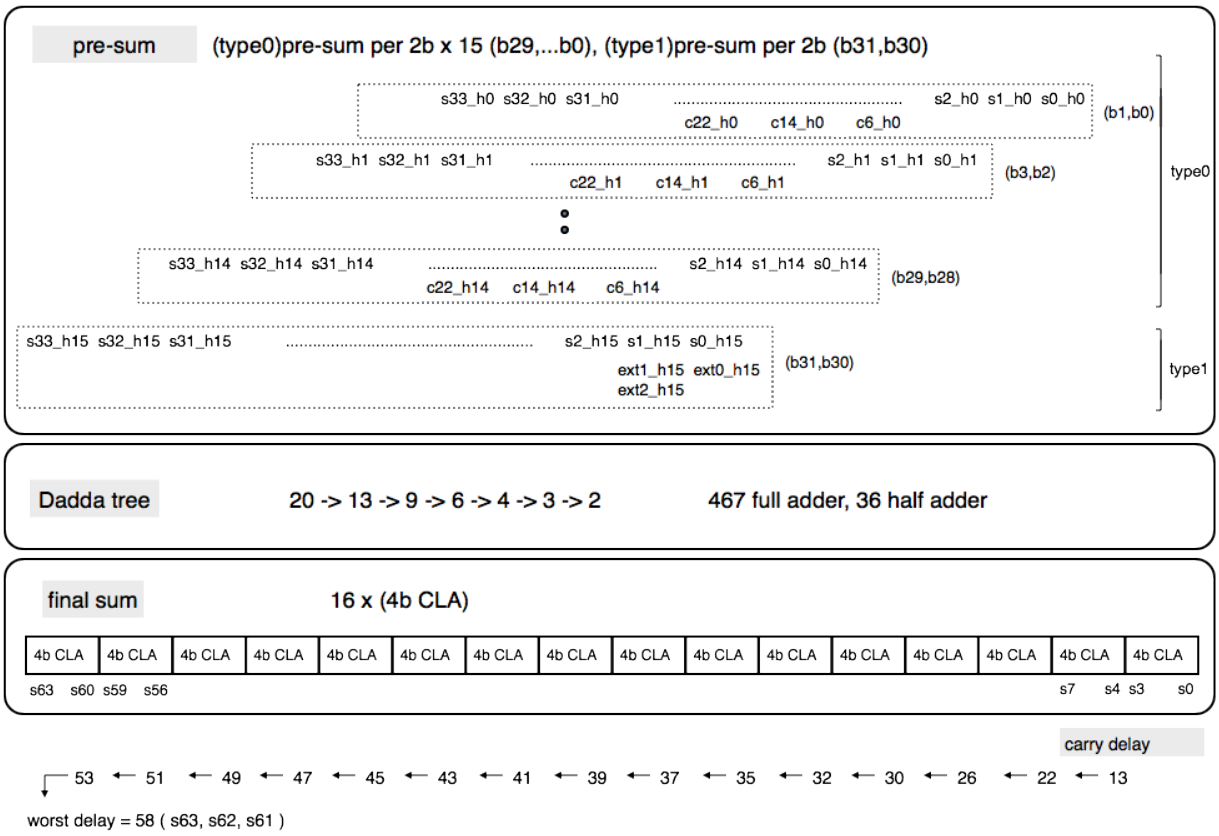
The worst delay is 58(s63,s62,s61).





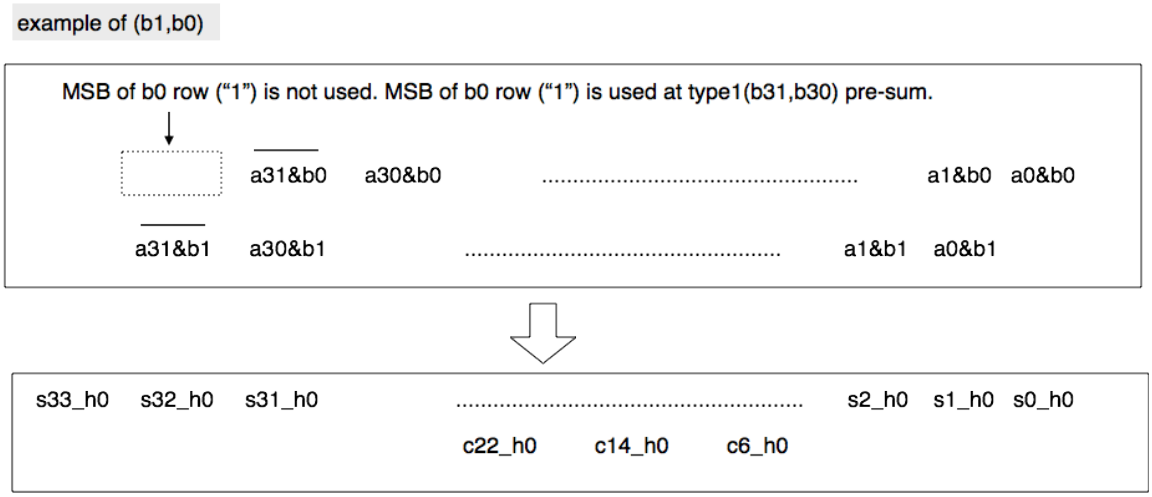
2 “pre-sum before Dadda tree using Baugh-Wooley” multiplier

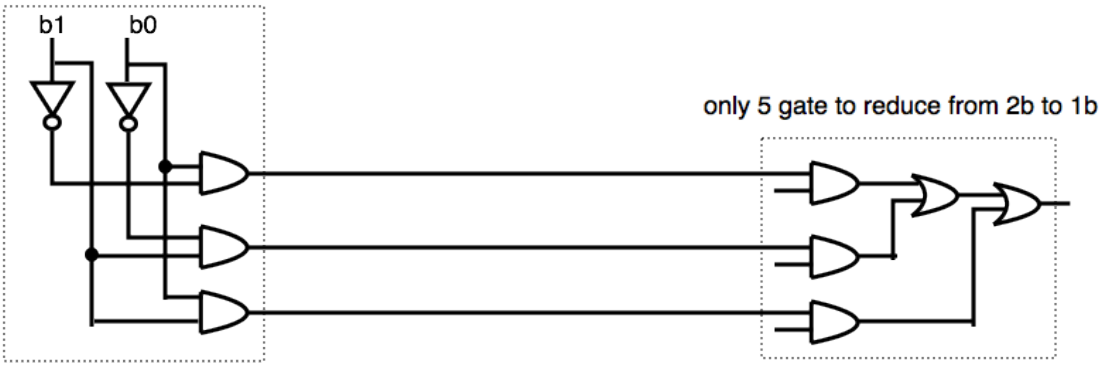
2.1 multiplier diagram



2.2 pre-sum

2.2.1 (type0) b29 to b0





the circuit is shared by a31,a30,....a1,a0

b1=0,b0=0

```
                1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+          1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
-------------------------------------------------------------------------------------------------------------
   0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

b1=0,b0=1

```
                a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10 a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
+          1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
-------------------------------------------------------------------------------------------------------------
   0  1  a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10 a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
```

b1=1,b0=0

```
                1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+          a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10  a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
-------------------------------------------------------------------------------------------------------------
(a31&a30) (a31^a30) a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10  a9  a8  a7  a6  a5  a4  a3  a2  a1  a0  0
```

b1=1,b0=1

```
                a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10 a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
+          a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10  a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
-------------------------------------------------------------------------------------------------------------
s33 s32 s31 s30 s29 s28 s27 s26 s25 s24 s23 s22 s21 s20 s19 s18 s17 s16 s15 s14 s13 s12 s11 s10 s9  s8  s7  s6  s5  s4  s3  s2  s1  s0
```

The details of (s33 to s0) are the following figures.

(s33 to s0)

```
                a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10 a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
+          a31 a30 a29 a28 a27 a26 a25 a24 a23 a22 a21 a20 a19 a18 a17 a16 a15 a14 a13 a12 a11 a10  a9  a8  a7  a6  a5  a4  a3  a2  a1  a0
-------------------------------------------------------------------------------------------------------------
c31 c30 c29 c28  c27 c26 c25  c24 c23 c22 c21 c20 c19  c18 c17 c16 c15 c14 c13 c12  c11 c10  c9  c8  c7  c6  c5  c4  c3  c2  c1  c0        carry
```

```
                a31 a30 a29 a28 a27 a26 a25 a24    a23 a22 a21 a20 a19 a18 a17 a16    a15 a14 a13 a12 a11 a10 a9  a8    a7  a6  a5  a4  a3  a2  a1  a0
+          a31 a30 a29 a28 a27 a26 a25 a24 a23    a22 a21 a20 a19 a18 a17 a16 a15    a14 a13 a12 a11 a10  a9  a8  a7    a6  a5  a4  a3  a2  a1  a0
-------------------------------------------------------------------------------------------------------------
                                  c22                                c14                                c6
```

The adder is cut every 8bit to reduce the carry delay. c6,c14 and c22 go to Dadda tree.

```
a31 a30 a29 a28 a27 a26 a25 a24 | a23 a22 a21 a20 a19 a18 a17 a16 | a15 a14 a13 a12 a11 a10 a9  a8  | a7  a6  a5  a4  a3  a2  a1  a0
                                ⊕ exor
a31 a30 a29 a28 a27 a26 a25 a24 a23 | a22 a21 a20 a19 a18 a17 a16 a15 | a14 a13 a12 a11 a10  a9  a8  a7  | a6  a5  a4  a3  a2  a1  a0
                                ⊕ exor
c31 c30 c29 c28  c27 c26 c25  c24 c23 | c21 c20 c19 c18 c17 c16  c15 | c13 c12  c11 c10  c9  c8  c7  | c5  c4  c3  c2  c1  c0
-------------------------------------------------------------------------------------------------------------------------------------
s33 s32 s31 s30  s29 s28 s27 s26 s25 s24 | s23 s22 s21 s20 s19 s18 s17 s16 | s15 s14 s13  s12 s11 s10 s9  s8  | s7  s6  s5  s4  s3  s2  s1  s0
                                                                                                                                    sum
```

gate delay

```
1   c0=(a0&a1);
2   c1=(a0&a1)              |(a1&a2);
3   c2=(a0&a1&a3)           |(a1&a2)          |(a2&a3);
4   c3=(a0&a1&a3)           |(a1&a2&a4)       |(a2&a3)          |(a3&a4);
4   c4=(a0&a1&a3&a5)        |(a1&a2&a4)       |(a2&a3&a5)       |(a3&a4)      |(a4&a5);
5   c5=(a0&a1&a3&a5)        |(a1&a2&a4&a6)|(a2&a3&a5)       |(a3&a4&a6)|(a4&a5)      |(a5&a6);
5   c6=(a0&a1&a3&a5&a7)|(a1&a2&a4&a6)|(a2&a3&a5&a7)|(a3&a4&a6)|(a4&a5&a7)|(a5&a6)|(a6&a7);
    (gate count)   c0-c6 = 40

1   c7 =(a7&a8);
2   c8 =(a7&a8)             |(a8&a9);
3   c9 =(a7&a8&a10)         |(a8&a9)          |(a9&a10);
4   c10=(a7&a8&a10)         |(a8&a9&a11)      |(a9&a10)         |(a10&a11);
4   c11=(a7&a8&a10&a12)     |(a8&a9&a11)      |(a9&a10&a12)     |(a10&a11)        |(a11&a12);
5   c12=(a7&a8&a10&a12)     |(a8&a9&a11&a13)  |(a9&a10&a12)     |(a10&a11&a13)    |(a11&a12)    |(a12&a13);
5   c13=(a7&a8&a10&a12&a14)|(a8&a9&a11&a13)  |(a9&a10&a12&a14)|(a10&a11&a13)    |(a11&a12&a14)|(a12&a13)    |(a13&a14);
6   c14=(a7&a8&a10&a12&a14)|(a8&a9&a11&a13&a15)|(a9&a10&a12&a14)|(a10&a11&a13&a15)|(a11&a12&a14)|(a12&a13&a15)|(a13&a14)|(a14&a15);
    (gate count)   c7-c14 = 52

1   c15=(a15&a16);
2   c16=(a15&a16)           |(a16&a17);
3   c17=(a15&a16&a18)       |(a16&a17)       |(a17&a18);
4   c18=(a15&a16&a18)       |(a16&a17&a19)    |(a17&a18)        |(a18&a19);
4   c19=(a15&a16&a18&a20)   |(a16&a17&a19)    |(a17&a18&a20)    |(a18&a19)        |(a19&a20);
5   c20=(a15&a16&a18&a20)   |(a16&a17&a19&a21)|(a17&a18&a20)    |(a18&a19&a21)    |(a19&a20)    |(a20&a21);
5   c21=(a15&a16&a18&a20&a22)|(a16&a17&a19&a21)|(a17&a18&a20&a22)|(a18&a19&a21)    |(a19&a20&a22)|(a20&a21)    |(a21&a22);
6   c22=(a15&a16&a18&a20&a22)|(a16&a17&a19&a21&a23)|(a17&a18&a20&a22)|(a18&a19&a21&a23)|(a19&a20&a22)|(a20&a21&a23)|(a21&a22)|(a22&a23);
    (gate count)   c15-c22 = 52

1   c23=(a23&a24);
2   c24=(a23&a24)           |(a24&a25);
3   c25=(a23&a24&a26)       |(a24&a25)       |(a25&a26);
4   c26=(a23&a24&a26)       |(a24&a25&a27)   |(a25&a26)        |(a26&a27);
4   c27=(a23&a24&a26&a28)   |(a24&a25&a27)   |(a25&a26&a28)    |(a26&a27)        |(a27&a28);
5   c28=(a23&a24&a26&a28)   |(a24&a25&a27&a29)|(a25&a26&a28)   |(a26&a27&a29)    |(a27&a28)    |(a28&a29);
5   c29=(a23&a24&a26&a28&a30)|(a24&a25&a27&a29)|(a25&a26&a28&a30)|(a26&a27&a29)   |(a27&a28&a30)|(a28&a29)    |(a29&a30);

6   c30=(a23&a24&a26&a28&a30)|(a24&a25&a27&a29&~a31)|(a25&a26&a28&a30)|(a26&a27&a29&~a31)|(a27&a28&a30)|(a28&a29&~a31)|(a29&a30)|(a30&~a31);

5   c31=                    (a24&a25&a27&a29&~a31)          |(a26&a27&a29&~a31)          |(a28&a29&~a31)       |(a30&~a31);
    (gate count)   c23-c31 = 55

    (total gate count ) 40 + 52 + 52 + 55 = 199
```

	gate delay	gate count
s0 = a0;	1	0
s1 = (a0^a1);	3	4
s2 =c0^(a1^a2);	6	8
s3 =c1^(a2^a3);	6	8
s4 =c2^(a3^a4);	6	8
s5 =c3^(a4^a5);	7	8
s6 =c4^(a5^a6);	7	8
s7 =c5^(a6^a7);	8	8
s8 = (a7^a8);	3	4
s9 = c7^(a8^a9);	6	8
s10= c8^(a9^a10);	6	8
s11= c9^(a10^a11);	6	8
s12=c10^(a11^a12);	7	8
s13=c11^(a12^a13);	7	8
s14=c12^(a13^a14);	8	8
s15=c13^(a14^a15);	8	8
s16= (a15^a16);	3	4
s17=c15^(a16^a17);	6	8
s18=c16^(a17^a18);	6	8
s19=c17^(a18^a19);	6	8
s20=c18^(a19^a20);	7	8
s21=c19^(a20^a21);	7	8
s22=c20^(a21^a22);	8	8
s23=c21^(a22^a23);	8	8
s24= (a23^a24);	3	4
s25=c23^(a24^a25);	6	8
s26=c24^(a25^a26);	6	8
s27=c25^(a26^a27);	6	8
s28=c26^(a27^a28);	7	8
s29=c27^(a28^a29);	7	8
s30=c28^(a29^a30);	8	8
s31=c29^(a30^(!a31));	8	8
s32=c30^(!a31);	9	4
s33=c31;	5	0
		(tota1 gate count) 236

The total gate count of (s33 to s0) = 199 + 236 = 435

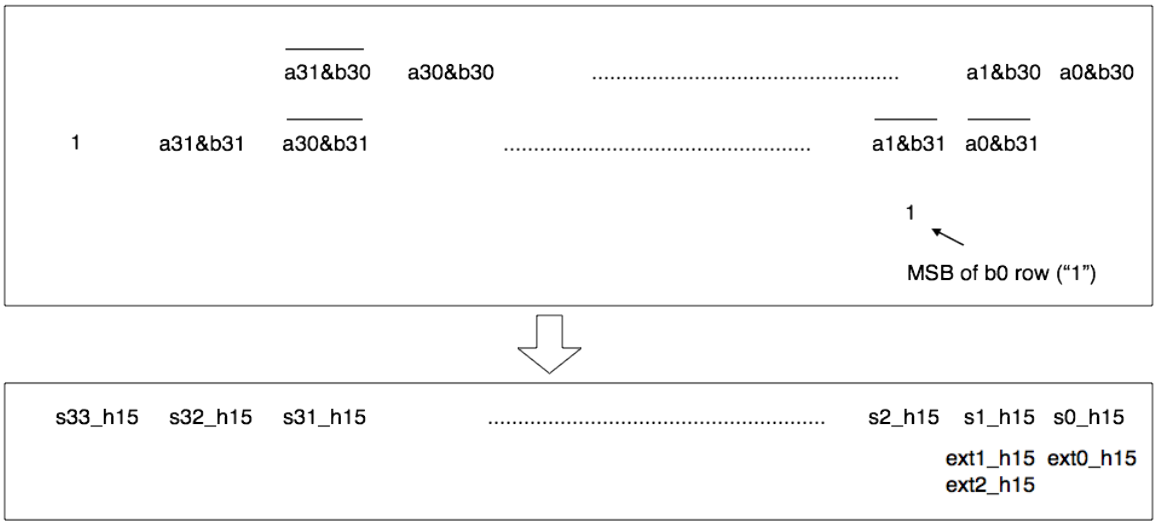
	gate delay	gate count
h0_11 = b1&b0;	1	1
h0_10 = b1&(!b0);	2	2
h0_01 = (!b1)&b0;	2	2
h0_00 = (!b1)&(!b0);	2	1
s0_h0 = (h0_11&s0) |(h0_01&a0);	4	3
s1_h0 = (h0_11&s1) |(h0_01&a1) |(h0_10&a0);	5	5
s2_h0 = (h0_11&s2) |(h0_01&a2) |(h0_10&a1);	8	5
s3_h0 = (h0_11&s3) |(h0_01&a3) |(h0_10&a2);	8	5
s4_h0 = (h0_11&s4) |(h0_01&a4) |(h0_10&a3);	8	5
s5_h0 = (h0_11&s5) |(h0_01&a5) |(h0_10&a4);	9	5
s6_h0 = (h0_11&s6) |(h0_01&a6) |(h0_10&a5);	9	5
s7_h0 = (h0_11&s7) |(h0_01&a7) |(h0_10&a6);	10	5
s8_h0 = (h0_11&s8) |(h0_01&a8) |(h0_10&a7);	5	5
s9_h0 = (h0_11&s9) |(h0_01&a9) |(h0_10&a8);	8	5
s10_h0 = (h0_11&s10)|(h0_01&a10)|(h0_10&a9);	8	5
s11_h0 = (h0_11&s11)|(h0_01&a11)|(h0_10&a10);	8	5
s12_h0 = (h0_11&s12)|(h0_01&a12)|(h0_10&a11);	9	5
s13_h0 = (h0_11&s13)|(h0_01&a13)|(h0_10&a12);	9	5
s14_h0 = (h0_11&s14)|(h0_01&a14)|(h0_10&a13);	10	5
s15_h0 = (h0_11&s15)|(h0_01&a15)|(h0_10&a14);	10	5
s16_h0 = (h0_11&s16)|(h0_01&a16)|(h0_10&a15);	5	5
s17_h0 = (h0_11&s17)|(h0_01&a17)|(h0_10&a16);	8	5
s18_h0 = (h0_11&s18)|(h0_01&a18)|(h0_10&a17);	8	5
s19_h0 = (h0_11&s19)|(h0_01&a19)|(h0_10&a18);	8	5
s20_h0 = (h0_11&s20)|(h0_01&a20)|(h0_10&a19);	9	5
s21_h0 = (h0_11&s21)|(h0_01&a21)|(h0_10&a20);	9	5
s22_h0 = (h0_11&s22)|(h0_01&a22)|(h0_10&a21);	10	5
s23_h0 = (h0_11&s23)|(h0_01&a23)|(h0_10&a22);	10	5
s24_h0 = (h0_11&s24)|(h0_01&a24)|(h0_10&a23);	5	5
s25_h0 = (h0_11&s25)|(h0_01&a25)|(h0_10&a24);	8	5
s26_h0 = (h0_11&s26)|(h0_01&a26)|(h0_10&a25);	8	5
s27_h0 = (h0_11&s27)|(h0_01&a27)|(h0_10&a26);	8	5
s28_h0 = (h0_11&s28)|(h0_01&a28)|(h0_10&a27);	9	5
s29_h0 = (h0_11&s29)|(h0_01&a29)|(h0_10&a28);	9	5
s30_h0 = (h0_11&s30)|(h0_01&a30)|(h0_10&a29);	10	5
s31_h0 = (h0_11&s31)|(h0_01&(!a31))|(h0_10&(!a30)) |(h0_00&1);	10	7
s32_h0 = (h0_11&s32)|(h0_01&1) |(h0_10&(a30^(!a31)))|(h0_00&1);	11	7
s33_h0 = (h0_11&s33)| |(h0_10&(a30&(!a31)));	6	3
c6_h0 = c6&h0_11;	6	1
c14_h0 = c14&h0_11;	7	1
c22_h0 = c22&h0_11;	7	1
		(total gate count) 179

The total gate count of (b1,b0) = 179

179 x 15 = 2,685 (b29,....b0)

2.2.2 (type1) b31,b30

(b31,b30)



	gate delay	gate count
h15_11 = b31&b30;	1	1
h15_10 = b31&(!b30);	2	2
h15_01 = (!b31)&b30;	2	2
h15_00 = (!b31)&(!b30);	2	1
s0_h15 = (h15_11&(!a0)) |(h15_01&a0);	4	3
s1_h15 = (h15_11&(!a1)) |(h15_01&a1) |(h15_10&(!a0));	5	5
s2_h15 = (h15_11&(!a2)) |(h15_01&a2) |(h15_10&(!a1));	5	5
s3_h15 = (h15_11&(!a3)) |(h15_01&a3) |(h15_10&(!a2));	5	5
s4_h15 = (h15_11&(!a4)) |(h15_01&a4) |(h15_10&(!a3));	5	5
s5_h15 = (h15_11&(!a5)) |(h15_01&a5) |(h15_10&(!a4));	5	5
s6_h15 = (h15_11&(!a6)) |(h15_01&a6) |(h15_10&(!a5));	5	5
s7_h15 = (h15_11&(!a7)) |(h15_01&a7) |(h15_10&(!a6));	5	5
s8_h15 = (h15_11&(!a8)) |(h15_01&a8) |(h15_10&(!a7));	5	5
s9_h15 = (h15_11&(!a9)) |(h15_01&a9) |(h15_10&(!a8));	5	5
s10_h15 = (h15_11&(!a10))|(h15_01&a10)|(h15_10&(!a9));	5	5
s11_h15 = (h15_11&(!a11))|(h15_01&a11)|(h15_10&(!a10));	5	5
s12_h15 = (h15_11&(!a12))|(h15_01&a12)|(h15_10&(!a11));	5	5
s13_h15 = (h15_11&(!a13))|(h15_01&a13)|(h15_10&(!a12));	5	5
s14_h15 = (h15_11&(!a14))|(h15_01&a14)|(h15_10&(!a13));	5	5
s15_h15 = (h15_11&(!a15))|(h15_01&a15)|(h15_10&(!a14));	5	5
s16_h15 = (h15_11&(!a16))|(h15_01&a16)|(h15_10&(!a15));	5	5
s17_h15 = (h15_11&(!a17))|(h15_01&a17)|(h15_10&(!a16));	5	5
s18_h15 = (h15_11&(!a18))|(h15_01&a18)|(h15_10&(!a17));	5	5
s19_h15 = (h15_11&(!a19))|(h15_01&a19)|(h15_10&(!a18));	5	5
s20_h15 = (h15_11&(!a20))|(h15_01&a20)|(h15_10&(!a19));	5	5
s21_h15 = (h15_11&(!a21))|(h15_01&a21)|(h15_10&(!a20));	5	5
s22_h15 = (h15_11&(!a22))|(h15_01&a22)|(h15_10&(!a21));	5	5
s23_h15 = (h15_11&(!a23))|(h15_01&a23)|(h15_10&(!a22));	5	5
s24_h15 = (h15_11&(!a24))|(h15_01&a24)|(h15_10&(!a23));	5	5
s25_h15 = (h15_11&(!a25))|(h15_01&a25)|(h15_10&(!a24));	5	5
s26_h15 = (h15_11&(!a26))|(h15_01&a26)|(h15_10&(!a25));	5	5
s27_h15 = (h15_11&(!a27))|(h15_01&a27)|(h15_10&(!a26));	5	5
s28_h15 = (h15_11&(!a28))|(h15_01&a28)|(h15_10&(!a27));	5	5
s29_h15 = (h15_11&(!a29))|(h15_01&a29)|(h15_10&(!a28));	5	5
s30_h15 = (h15_11&(!a30))|(h15_01&a30)|(h15_10&(!a29));	5	5
s31_h15 = (h15_11&a31)|(h15_01&(!a31))|(h15_10&a30) |(h15_00&1);	5	7
s32_h15 = (h15_11&1) |(h15_01&1) |(h15_10&((!a30)^a31)) |(h15_00&1);	5	7
s33_h15 = (h15_11&1) |(h15_01&1) |(h15_10&(!((!a30)&a31)))|(h15_00&1);	5	7
ex0_h15 = h15_11;	1	0
ex1_h15 = 1;	1	0
ex2_h15 = h15_10;	2	0
		(total gate count) 180

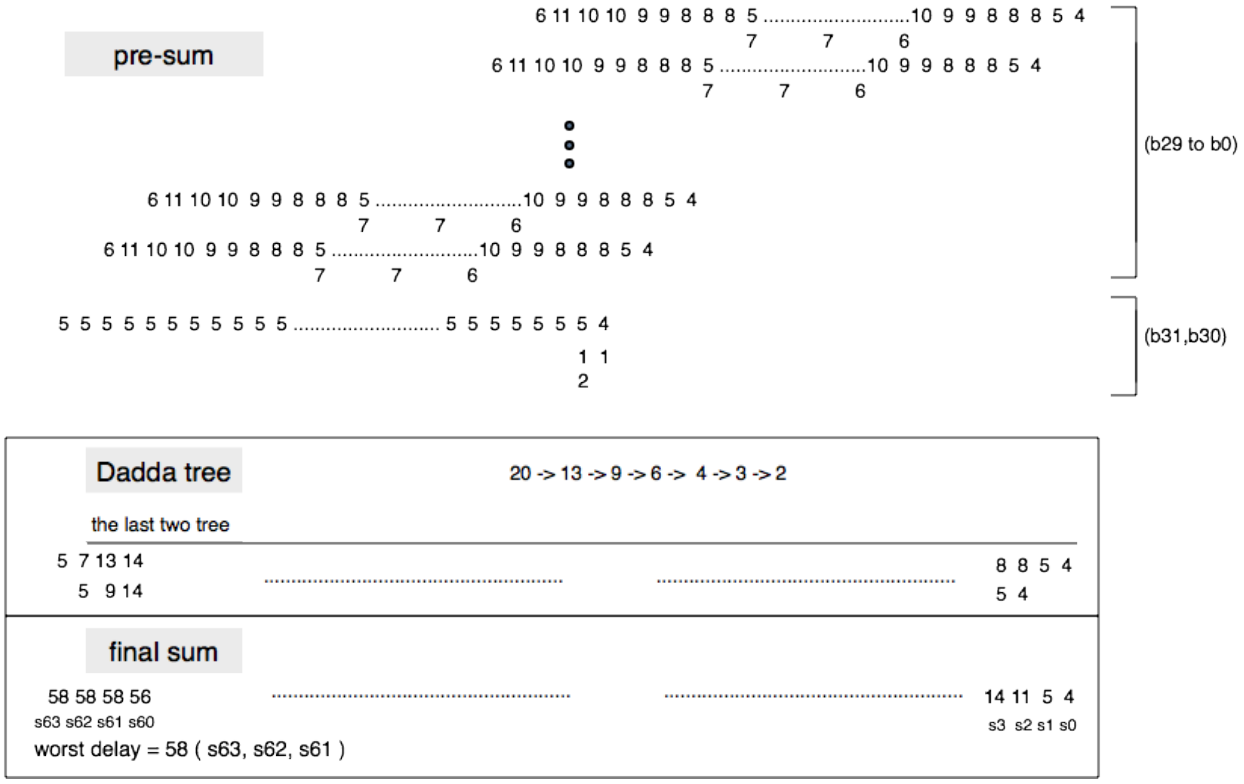
The total gate count of (b31,b30) = 180

2.3 gate count

```
[1] pre-sum
(a31,a30,..a1,a0) x 3 = 236 + 199 = 435
(b0,b1,...b29) pre-sum per 2b x 15 = 179 x 15 = 2,685
(b30,b31)      pre-sum per 2b = 180

[2] Dadda tree
full adder(= 10 gate count) x 467 = 4,670
half adder(= 4 gate count) x 36 = 144

[3] final sum
4bCLA adder(= 60 gate count) x 16 = 960

[4] total gate count
435 + 2,685 + 180 + 4,670 + 144 + 960 = 9,074
```

2.4 gate delay

The worst delay is 58(s63,s62,s61).



3 Dadda tree calculation

[1] Dadda tree reduction program

g++ -o daddatree daddatree.cpp

[example]

9 input   3 full adder

input.txt
15 11 12 15 13 12 12 13 12 Z

```
-----------------------------------------------
%./daddatree  input.txt  3

bufinlen0 :9
15 11 12 15 13 12 12 13 12
bufinlen1 :9
15 15 13 13 12 12 12 12 11
bufinlen2 :9
15000 15000 13000 13000 12000 12000 12000 12000 11000
bigger
no thru
resultthru s0
no half adder
resultha c0
resultha s0
first_val:15 second_val:13,third_val:13
resultfa c17
resultfa s19
first_val:15 second_val:12,third_val:12
resultfa c17
resultfa s18
first_val:12 second_val:12,third_val:11
resultfa c15
resultfa s18
sum 3 carry 3
FINAL sumbuf 19 18 18
FINAL carrybuf 17 17 15
```

```
//daddatree.cpp


//--------------------------------------------------------------------------------
// 1st     release   Dec.24,2023 : all codes from scratch
//--------------------------------------------------------------------------------
//--------------------------------------------------------------------------------
// version 1.0 Dec.24,2023
//--------------------------------------------------------------------------------
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <string>
#include <math.h>
using namespace std;
void daddasort(int bufinlen,int *bufin);
void daddafa(int first_val,int second_val,int third_val);
void daddatreecalc(int *mainbuf,int afternum);

int main(int argc, char * const argv[]) {

  int  mainbuf[1024];//equation main
        char c0;
        int  d0;
        int  i;
        int  afternum;

        ifstream fv0(argv[1]);//equation

        for(i=0;i<1024;i++){
                mainbuf[i]=0;
        }
        for(i=0;i<1024;i++){
                fv0.get(c0);
                d0=(int)c0;
                mainbuf[i]=d0;
                if(d0==90){break;}
        }

        afternum  = atoi(argv[2]);
```

```
        daddatreecalc(mainbuf,afternum);

        fv0.close();
        return 0;
}

//---------------------------------
//function : daddatreecalculation
// 3bit x n -> 2bit x n
//---------------------------------
void daddatreecalc(int *mainbuf,int afternum){

        int i,j,m,n;
        int bufin[256];
        int bufinlen;
        int flag;
        int first_val,second_val,third_val;
        int daddavalue;
        int half_carry,half_sum;
        int carrybuf[256];
        int sumbuf[256];
        int fanum,hanum;

        //-----------------------------
        // store the value in bufin[].
        //-----------------------------
        m=0;
        daddavalue=0;
        for(i=0;i<32768;i++){
                if(mainbuf[i]==90){ // char "Z" = 90
                        break;
                }
                else if(mainbuf[i]==32 & i==0){ // 20230510 ignore the 1st char " " = 32
                }
                else if(mainbuf[i]==32){ // char " " = 32
                        bufin[m]=daddavalue;
                        m++;
                        daddavalue=0;
                }
                else{
                        daddavalue=daddavalue*10;
                        daddavalue=daddavalue+(mainbuf[i]-48);//char "0" = 48
                }
        }
        bufinlen=m;

        //-----------------------------
        // fanum hanum calculation
        //-----------------------------
        fanum = (bufinlen -afternum)/2;
        hanum = (bufinlen -afternum)%2;

        //-----------------------------
        // calculation
        //-----------------------------
        printf("bufinlen0 :%d\n",bufinlen);
        for(i=0; i<bufinlen; i++) {
                printf("%d ",bufin[i]);
        }
        printf("\n");

        daddasort(bufinlen, bufin);

        printf("bufinlen1 :%d\n",bufinlen);
        for(i=0; i<bufinlen; i++) {
                printf("%d ",bufin[i]);
        }
        printf("\n");

        flag=0;

        for(i=0;i<bufinlen;i++){
                if(i>=(bufinlen-fanum*3)){
                        bufin[i]=bufin[i]*1000;
                }
                else{
                        bufin[i]=bufin[i];
                }
        }

        printf("bufinlen2 :%d\n",bufinlen);
        for(i=0; i<bufinlen; i++) {
                printf("%d ",bufin[i]);
        }
        printf("\n");

//-----------------------------------
// bufinlen is smaller then afternum?
//-----------------------------------
//smaller
if(bufinlen<=afternum){
          printf("smaller\n");
          m=0;
                n=0;
          for(i=0;i<bufinlen;i++){
                  sumbuf[m]=bufin[i]; m++;
                }
          printf("\n");
```

```
  printf("smaller m n: %d %d\n",m,n);
        printf("FINAL sumbuf ");
        for(i=0;i<m;i++){
                printf("%d ",sumbuf[i]);
        }
        printf("\n");

        printf("FINAL carrybuf ");
        if(n==0){
          printf("0");
        }
        else{
          for(i=0;i<n;i++){
                   printf("%d ",carrybuf[i]);
          }
        }
        printf("\n");
}

//bigger
else{
        printf("bigger\n");

  m=0;
        n=0;
  //-----------------
  //thru
  //-----------------
        if((bufinlen-fanum*3-hanum*2)>0){
          for(i=0;i<(bufinlen-fanum*3-hanum*2);i++){
                printf("resultthru s%d\n",bufin[i]);
                        sumbuf[m]=bufin[i]; m++;
                }
          printf("\n");
        }
        else{
                printf("no thru\n");
          printf("resultthru s0\n");
        }

  //-----------------
  //half adder
  //-----------------
        if((bufinlen-fanum*3-hanum*2)>=0 & hanum==1){//20230511
          printf("half adder %d,%d\n",bufin[bufinlen-fanum*3-hanum*2],bufin[bufinlen-fanum*3-hanum*2+1]);
          half_carry=bufin[bufinlen-fanum*3-hanum*2]+1;
          half_sum  =bufin[bufinlen-fanum*3-hanum*2]+3;
          printf("resultha c%d\nresultha s%d\n",half_carry,half_sum);
                sumbuf[m]=half_sum;   m++;
                carrybuf[n]=half_carry; n++;
        }
        else{
                printf("no half adder\n");
          printf("resultha c0\nresultha s0\n");
        }

  //-----------------
  //full adder
  //-----------------
        for(i=0;i<fanum;i++){
                daddasort(bufinlen, bufin);
                flag=0;

                for(j=0;j<bufinlen;j++){
                        if(j==0){
                                bufin[j]=bufin[j]/1000;
                                first_val=bufin[j];
                        }
                        else if(j>0 & j<(fanum-i)){
                                if((bufin[j]/1000<=(bufin[0]-3)) & flag==0){
                                        flag=1;
                                        bufin[j]=bufin[j]/1000;
                                        bufin[j+1]=bufin[j+1]/1000;
                                        second_val=bufin[j];
                                        third_val=bufin[j+1];
                                }
                        }
                        else if(j==(fanum-i)){
                                if(flag==0){
                                        if(bufin[j]==bufin[j-1]){
                                                flag=1;
                                                bufin[j]=bufin[j]/1000;
                                                bufin[j-1]=bufin[j-1]/1000;
                                                second_val=bufin[j];
                                                third_val=bufin[j-1];
                                        }
                                        else{
                                                flag=1;
                                                bufin[j]=bufin[j]/1000;
                                                bufin[j+1]=bufin[j+1]/1000;
                                                second_val=bufin[j];
                                                third_val=bufin[j+1];
                                        }
                                }
                        }
                }
                printf("first_val:%d second_val:%d,third_val:%d\n",first_val,second_val,third_val);

          int first_carry,first_sum;
```

```
          int second_carry,second_sum;

          first_carry=first_val+2;
          first_sum=first_val+3;
          second_carry=second_val+3;
          second_sum=second_val+6;

          if(first_carry>=second_carry){
                  printf("resultfa c%d\n",first_carry);
                  carrybuf[n]=first_carry; n++;
          }
          else{
                  printf("resultfa c%d\n",second_carry);
                  carrybuf[n]=second_carry; n++;
          }

          if(first_sum>=second_sum){
                  printf("resultfa s%d\n",first_sum);
                  sumbuf[m]=first_sum;   m++;
          }
          else{
                  printf("resultfa s%d\n",second_sum);
                  sumbuf[m]=second_sum;   m++;
          }
        }

  printf("sum %d carry %d\n",m,n);

        printf("FINAL sumbuf ");
        for(i=0;i<m;i++){
                printf("%d ",sumbuf[i]);
        }
        printf("\n");

        printf("FINAL carrybuf ");
        if(n==0){
          printf("0");
        }
        else{
          for(i=0;i<n;i++){
                  printf("%d ",carrybuf[i]);
          }
        }
        printf("\n");
}
}


void daddasort(int bufinlen,int *bufin){

        int i,j,k;
        int min,tmp;
        int bufintmp[256];

        for(i=0;i<bufinlen-1;i++){
                min=bufin[i];
                k=i;
                for(j=i+1;j<bufinlen;j++){
                        if(bufin[j]<min){
                                min=bufin[j];
                                k=j;
                        }
                }
                tmp=bufin[i];
                bufin[i]=bufin[k];
                bufin[k]=tmp;
        }

        //sort reverse
        for(i=0;i<bufinlen;i++){
                bufintmp[bufinlen-1-i]=bufin[i];
        }
        for(i=0;i<bufinlen;i++){
                bufin[i]=bufintmp[i];
        }
}
```

[2] “pre-sum before Dadda tree using Baugh-Wooley” simulation model

```
g++ -o 32bx32b_signed_baughwooley_presum_simulation_model
32bx32b_signed_baughwooley_presum_simulation_model.cpp

./32bx32b_signed_baughwooley_presum_simulation_model
```

```
//32bx32b_signed_baughwooley_presum_simulation_model.cpp


//--------------------------------------------------------------------------------
// 1st     release   Dec.26,2023 :
//--------------------------------------------------------------------------------
//--------------------------------------------------------------------------------
// version 1.0 Dec.26,2023
// 32bx32b signed baugh-wooley presum simulation model
//--------------------------------------------------------------------------------
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <string>
#include <math.h>
using namespace std;

  int main(int argc, char * const argv[]) {

  long int i,j,k;
  long int aa0,bb0;
  long int ain,bin,sout;
  long int expected_sout;
  long int sout2;

  int pata[32],patb[32];
  int a0, a1, a2, a3, a4, a5, a6, a7;
  int a8, a9, a10,a11,a12,a13,a14,a15;
  int a16,a17,a18,a19,a20,a21,a22,a23;
  int a24,a25,a26,a27,a28,a29,a30,a31;
  int b0, b1, b2, b3, b4, b5, b6, b7;
  int b8, b9, b10,b11,b12,b13,b14,b15;
  int b16,b17,b18,b19,b20,b21,b22,b23;
  int b24,b25,b26,b27,b28,b29,b30,b31;

  int h0_00,h0_01,h0_10,h0_11;
  int h1_00,h1_01,h1_10,h1_11;
  int h2_00,h2_01,h2_10,h2_11;
  int h3_00,h3_01,h3_10,h3_11;
  int h4_00,h4_01,h4_10,h4_11;
  int h5_00,h5_01,h5_10,h5_11;
  int h6_00,h6_01,h6_10,h6_11;
  int h7_00,h7_01,h7_10,h7_11;
  int h8_00,h8_01,h8_10,h8_11;
  int h9_00,h9_01,h9_10,h9_11;
  int h10_00,h10_01,h10_10,h10_11;
  int h11_00,h11_01,h11_10,h11_11;
  int h12_00,h12_01,h12_10,h12_11;
  int h13_00,h13_01,h13_10,h13_11;
  int h14_00,h14_01,h14_10,h14_11;
  int h15_00,h15_01,h15_10,h15_11;

  int pre_c0,pre_c1,pre_c2,pre_c3,pre_c4,pre_c5,pre_c6,pre_c7;
  int pre_c8,pre_c9,pre_c10,pre_c11,pre_c12,pre_c13,pre_c14,pre_c15;
  int pre_c16,pre_c17,pre_c18,pre_c19,pre_c20,pre_c21,pre_c22,pre_c23;
  int pre_c24,pre_c25,pre_c26,pre_c27,pre_c28,pre_c29,pre_c30,pre_c31;

  int cut0_h0,cut0_h1,cut0_h2,cut0_h3,cut0_h4,cut0_h5,cut0_h6,cut0_h7;
  int cut0_h8,cut0_h9,cut0_h10,cut0_h11,cut0_h12,cut0_h13,cut0_h14;
  int cut1_h0,cut1_h1,cut1_h2,cut1_h3,cut1_h4,cut1_h5,cut1_h6,cut1_h7;
  int cut1_h8,cut1_h9,cut1_h10,cut1_h11,cut1_h12,cut1_h13,cut1_h14;
  int cut2_h0,cut2_h1,cut2_h2,cut2_h3,cut2_h4,cut2_h5,cut2_h6,cut2_h7;
  int cut2_h8,cut2_h9,cut2_h10,cut2_h11,cut2_h12,cut2_h13,cut2_h14;

  int pre_s0,pre_s1,pre_s2,pre_s3,pre_s4,pre_s5,pre_s6,pre_s7;
  int pre_s8,pre_s9,pre_s10,pre_s11,pre_s12,pre_s13,pre_s14,pre_s15;
  int pre_s16,pre_s17,pre_s18,pre_s19,pre_s20,pre_s21,pre_s22,pre_s23;
  int pre_s24,pre_s25,pre_s26,pre_s27,pre_s28,pre_s29,pre_s30,pre_s31;
  int pre_s32,pre_s33;

  int s0_h0,s1_h0,s2_h0,s3_h0,s4_h0,s5_h0,s6_h0,s7_h0,s8_h0,s9_h0;
  int s10_h0,s11_h0,s12_h0,s13_h0,s14_h0,s15_h0,s16_h0,s17_h0;
  int s18_h0,s19_h0,s20_h0,s21_h0,s22_h0,s23_h0,s24_h0,s25_h0;
  int s26_h0,s27_h0,s28_h0,s29_h0,s30_h0,s31_h0,s32_h0,s33_h0;

  int s0_h1,s1_h1,s2_h1,s3_h1,s4_h1,s5_h1,s6_h1,s7_h1,s8_h1,s9_h1;
  int s10_h1,s11_h1,s12_h1,s13_h1,s14_h1,s15_h1,s16_h1,s17_h1;
  int s18_h1,s19_h1,s20_h1,s21_h1,s22_h1,s23_h1,s24_h1,s25_h1;
  int s26_h1,s27_h1,s28_h1,s29_h1,s30_h1,s31_h1,s32_h1,s33_h1;

  int s0_h2,s1_h2,s2_h2,s3_h2,s4_h2,s5_h2,s6_h2,s7_h2,s8_h2,s9_h2;
  int s10_h2,s11_h2,s12_h2,s13_h2,s14_h2,s15_h2,s16_h2,s17_h2;
  int s18_h2,s19_h2,s20_h2,s21_h2,s22_h2,s23_h2,s24_h2,s25_h2;
  int s26_h2,s27_h2,s28_h2,s29_h2,s30_h2,s31_h2,s32_h2,s33_h2;

  int s0_h3,s1_h3,s2_h3,s3_h3,s4_h3,s5_h3,s6_h3,s7_h3,s8_h3,s9_h3;
  int s10_h3,s11_h3,s12_h3,s13_h3,s14_h3,s15_h3,s16_h3,s17_h3;
```

```
int s18_h3,s19_h3,s20_h3,s21_h3,s22_h3,s23_h3,s24_h3,s25_h3;
int s26_h3,s27_h3,s28_h3,s29_h3,s30_h3,s31_h3,s32_h3,s33_h3;

int s0_h4,s1_h4,s2_h4,s3_h4,s4_h4,s5_h4,s6_h4,s7_h4,s8_h4,s9_h4;
int s10_h4,s11_h4,s12_h4,s13_h4,s14_h4,s15_h4,s16_h4,s17_h4;
int s18_h4,s19_h4,s20_h4,s21_h4,s22_h4,s23_h4,s24_h4,s25_h4;
int s26_h4,s27_h4,s28_h4,s29_h4,s30_h4,s31_h4,s32_h4,s33_h4;

int s0_h5,s1_h5,s2_h5,s3_h5,s4_h5,s5_h5,s6_h5,s7_h5,s8_h5,s9_h5;
int s10_h5,s11_h5,s12_h5,s13_h5,s14_h5,s15_h5,s16_h5,s17_h5;
int s18_h5,s19_h5,s20_h5,s21_h5,s22_h5,s23_h5,s24_h5,s25_h5;
int s26_h5,s27_h5,s28_h5,s29_h5,s30_h5,s31_h5,s32_h5,s33_h5;

int s0_h6,s1_h6,s2_h6,s3_h6,s4_h6,s5_h6,s6_h6,s7_h6,s8_h6,s9_h6;
int s10_h6,s11_h6,s12_h6,s13_h6,s14_h6,s15_h6,s16_h6,s17_h6;
int s18_h6,s19_h6,s20_h6,s21_h6,s22_h6,s23_h6,s24_h6,s25_h6;
int s26_h6,s27_h6,s28_h6,s29_h6,s30_h6,s31_h6,s32_h6,s33_h6;

int s0_h7,s1_h7,s2_h7,s3_h7,s4_h7,s5_h7,s6_h7,s7_h7,s8_h7,s9_h7;
int s10_h7,s11_h7,s12_h7,s13_h7,s14_h7,s15_h7,s16_h7,s17_h7;
int s18_h7,s19_h7,s20_h7,s21_h7,s22_h7,s23_h7,s24_h7,s25_h7;
int s26_h7,s27_h7,s28_h7,s29_h7,s30_h7,s31_h7,s32_h7,s33_h7;

int s0_h8,s1_h8,s2_h8,s3_h8,s4_h8,s5_h8,s6_h8,s7_h8,s8_h8,s9_h8;
int s10_h8,s11_h8,s12_h8,s13_h8,s14_h8,s15_h8,s16_h8,s17_h8;
int s18_h8,s19_h8,s20_h8,s21_h8,s22_h8,s23_h8,s24_h8,s25_h8;
int s26_h8,s27_h8,s28_h8,s29_h8,s30_h8,s31_h8,s32_h8,s33_h8;

int s0_h9,s1_h9,s2_h9,s3_h9,s4_h9,s5_h9,s6_h9,s7_h9,s8_h9,s9_h9;
int s10_h9,s11_h9,s12_h9,s13_h9,s14_h9,s15_h9,s16_h9,s17_h9;
int s18_h9,s19_h9,s20_h9,s21_h9,s22_h9,s23_h9,s24_h9,s25_h9;
int s26_h9,s27_h9,s28_h9,s29_h9,s30_h9,s31_h9,s32_h9,s33_h9;

int s0_h10,s1_h10,s2_h10,s3_h10,s4_h10,s5_h10,s6_h10,s7_h10,s8_h10,s9_h10;
int s10_h10,s11_h10,s12_h10,s13_h10,s14_h10,s15_h10,s16_h10,s17_h10;
int s18_h10,s19_h10,s20_h10,s21_h10,s22_h10,s23_h10,s24_h10,s25_h10;
int s26_h10,s27_h10,s28_h10,s29_h10,s30_h10,s31_h10,s32_h10,s33_h10;

int s0_h11,s1_h11,s2_h11,s3_h11,s4_h11,s5_h11,s6_h11,s7_h11,s8_h11,s9_h11;
int s10_h11,s11_h11,s12_h11,s13_h11,s14_h11,s15_h11,s16_h11,s17_h11;
int s18_h11,s19_h11,s20_h11,s21_h11,s22_h11,s23_h11,s24_h11,s25_h11;
int s26_h11,s27_h11,s28_h11,s29_h11,s30_h11,s31_h11,s32_h11,s33_h11;

int s0_h12,s1_h12,s2_h12,s3_h12,s4_h12,s5_h12,s6_h12,s7_h12,s8_h12,s9_h12;
int s10_h12,s11_h12,s12_h12,s13_h12,s14_h12,s15_h12,s16_h12,s17_h12;
int s18_h12,s19_h12,s20_h12,s21_h12,s22_h12,s23_h12,s24_h12,s25_h12;
int s26_h12,s27_h12,s28_h12,s29_h12,s30_h12,s31_h12,s32_h12,s33_h12;

int s0_h13,s1_h13,s2_h13,s3_h13,s4_h13,s5_h13,s6_h13,s7_h13,s8_h13,s9_h13;
int s10_h13,s11_h13,s12_h13,s13_h13,s14_h13,s15_h13,s16_h13,s17_h13;
int s18_h13,s19_h13,s20_h13,s21_h13,s22_h13,s23_h13,s24_h13,s25_h13;
int s26_h13,s27_h13,s28_h13,s29_h13,s30_h13,s31_h13,s32_h13,s33_h13;

int s0_h14,s1_h14,s2_h14,s3_h14,s4_h14,s5_h14,s6_h14,s7_h14,s8_h14,s9_h14;
int s10_h14,s11_h14,s12_h14,s13_h14,s14_h14,s15_h14,s16_h14,s17_h14;
int s18_h14,s19_h14,s20_h14,s21_h14,s22_h14,s23_h14,s24_h14,s25_h14;
int s26_h14,s27_h14,s28_h14,s29_h14,s30_h14,s31_h14,s32_h14,s33_h14;

int s0_h15,s1_h15,s2_h15,s3_h15,s4_h15,s5_h15,s6_h15,s7_h15,s8_h15,s9_h15;
int s10_h15,s11_h15,s12_h15,s13_h15,s14_h15,s15_h15,s16_h15,s17_h15;
int s18_h15,s19_h15,s20_h15,s21_h15,s22_h15,s23_h15,s24_h15,s25_h15;
int s26_h15,s27_h15,s28_h15,s29_h15,s30_h15,s31_h15,s32_h15,s33_h15;

int ext0_h15,ext1_h15,ext2_h15;

long int sum0,sum1,sum2,sum3,sum4,sum5,sum6,sum7;
long int sum8,sum9,sum10,sum11,sum12,sum13,sum14,sum15;

long int pow00,pow01,pow02,pow03,pow04,pow05,pow06,pow07,pow08,pow09;
long int pow10,pow11,pow12,pow13,pow14,pow15,pow16,pow17,pow18,pow19;
long int pow20,pow21,pow22,pow23,pow24,pow25,pow26,pow27,pow28,pow29;
long int pow30,pow31,pow32,pow33,pow34;
long int pow63,pow64;

long int width;
long int r0,r1,r2;

pow00=(long int)pow(2,0);
pow01=(long int)pow(2,1);
pow02=(long int)pow(2,2);
pow03=(long int)pow(2,3);
pow04=(long int)pow(2,4);
pow05=(long int)pow(2,5);
pow06=(long int)pow(2,6);
pow07=(long int)pow(2,7);
pow08=(long int)pow(2,8);
pow09=(long int)pow(2,9);
pow10=(long int)pow(2,10);
pow11=(long int)pow(2,11);
pow12=(long int)pow(2,12);
pow13=(long int)pow(2,13);
pow14=(long int)pow(2,14);
pow15=(long int)pow(2,15);
pow16=(long int)pow(2,16);
pow17=(long int)pow(2,17);
pow18=(long int)pow(2,18);
pow19=(long int)pow(2,19);
pow20=(long int)pow(2,20);
pow21=(long int)pow(2,21);
```

```
  pow22=(long int)pow(2,22);
  pow23=(long int)pow(2,23);
  pow24=(long int)pow(2,24);
  pow25=(long int)pow(2,25);
  pow26=(long int)pow(2,26);
  pow27=(long int)pow(2,27);
  pow28=(long int)pow(2,28);
  pow29=(long int)pow(2,29);
  pow30=(long int)pow(2,30);
  pow31=(long int)pow(2,31);
  pow32=(long int)pow(2,32);
  pow33=(long int)pow(2,33);
  pow34=(long int)pow(2,34);
  pow63=(long int)pow(2,63);
  pow64=(long int)pow(2,64);

  unsigned int now=(unsigned int)time(NULL);
  srand(now);

  width = pow32;
  printf("rand : %d\n",rand());

  for (i=0; i<pow16   ; i++) {

    r0=rand();
    r1=rand();
    r2=r0*r1;
    ain=r2%width-width/2;

    r0=rand();
    r1=rand();
    r2=r0*r1;
    bin=r2%width-width/2;

    aa0=ain;
    bb0=bin;

    sout2=ain*bin;
    expected_sout=sout2;

    if(aa0<0){
    aa0=aa0+width;
    }
    if(bb0<0){
    bb0=bb0+width;
    }

    for (k=0; k<32; k++) {
    pata[k]=aa0%2;
    aa0=aa0/2;
    patb[k]=bb0%2;
    bb0=bb0/2;
    }

    a0=pata[0];
    b0=patb[0];
    a1=pata[1];
    b1=patb[1];
    a2=pata[2];
    b2=patb[2];
    a3=pata[3];
    b3=patb[3];
    a4=pata[4];
    b4=patb[4];
    a5=pata[5];
    b5=patb[5];
    a6=pata[6];
    b6=patb[6];
    a7=pata[7];
    b7=patb[7];
    a8=pata[8];
    b8=patb[8];
    a9=pata[9];
    b9=patb[9];
    a10=pata[10];
    b10=patb[10];
    a11=pata[11];
    b11=patb[11];
    a12=pata[12];
    b12=patb[12];
    a13=pata[13];
    b13=patb[13];
    a14=pata[14];
    b14=patb[14];
    a15=pata[15];
    b15=patb[15];
    a16=pata[16];
    b16=patb[16];
    a17=pata[17];
    b17=patb[17];
    a18=pata[18];
    b18=patb[18];
    a19=pata[19];
    b19=patb[19];
    a20=pata[20];
    b20=patb[20];
    a21=pata[21];
    b21=patb[21];
    a22=pata[22];
```

```
    b22=patb[22];
    a23=pata[23];
    b23=patb[23];
    a24=pata[24];
    b24=patb[24];
    a25=pata[25];
    b25=patb[25];
    a26=pata[26];
    b26=patb[26];
    a27=pata[27];
    b27=patb[27];
    a28=pata[28];
    b28=patb[28];
    a29=pata[29];
    b29=patb[29];
    a30=pata[30];
    b30=patb[30];
    a31=pata[31];
    b31=patb[31];

    //x3 carry(MSB negated)
    pre_c0=(a0&a1);
    pre_c1=(a0&a1)          |(a1&a2);
    pre_c2=(a0&a1&a3)       |(a1&a2)      |(a2&a3);
    pre_c3=(a0&a1&a3)       |(a1&a2&a4)   |(a2&a3)      |(a3&a4);
    pre_c4=(a0&a1&a3&a5)    |(a1&a2&a4)   |(a2&a3&a5)   |(a3&a4)   |(a4&a5);
    pre_c5=(a0&a1&a3&a5)    |(a1&a2&a4&a6)|(a2&a3&a5)   |(a3&a4&a6)|(a4&a5)   |(a5&a6);
    pre_c6=(a0&a1&a3&a5&a7)|(a1&a2&a4&a6)|(a2&a3&a5&a7)|(a3&a4&a6)|(a4&a5&a7)|(a5&a6)|(a6&a7);

    pre_c7 =(a7&a8);
    pre_c8 =(a7&a8)            |(a8&a9);
    pre_c9 =(a7&a8&a10)        |(a8&a9)            |(a9&a10);
    pre_c10=(a7&a8&a10)        |(a8&a9&a11)        |(a9&a10)        |(a10&a11);
    pre_c11=(a7&a8&a10&a12)    |(a8&a9&a11)        |(a9&a10&a12)    |(a10&a11)        |(a11&a12);
    pre_c12=(a7&a8&a10&a12)    |(a8&a9&a11&a13)    |(a9&a10&a12)    |(a10&a11&a13)    |(a11&a12)    |(a12&a13);
    pre_c13=(a7&a8&a10&a12&a14)|(a8&a9&a11&a13)    |(a9&a10&a12&a14)|(a10&a11&a13)    |(a11&a12&a14)|(a12&a13)
|(a13&a14);
    pre_c14=(a7&a8&a10&a12&a14)|(a8&a9&a11&a13&a15)|(a9&a10&a12&a14)|(a10&a11&a13&a15)|(a11&a12&a14)|
(a12&a13&a15) |(a13&a14) |(a14&a15);

    pre_c15=(a15&a16);
    pre_c16=(a15&a16)           |(a16&a17);
    pre_c17=(a15&a16&a18)       |(a16&a17)           |(a17&a18);
    pre_c18=(a15&a16&a18)       |(a16&a17&a19)       |(a17&a18)       |(a18&a19);
    pre_c19=(a15&a16&a18&a20)   |(a16&a17&a19)       |(a17&a18&a20)   |(a18&a19)       |(a19&a20);
    pre_c20=(a15&a16&a18&a20)   |(a16&a17&a19&a21)   |(a17&a18&a20)   |(a18&a19&a21)   |(a19&a20)   |
(a20&a21);
    pre_c21=(a15&a16&a18&a20&a22)|(a16&a17&a19&a21)   |(a17&a18&a20&a22)|(a18&a19&a21)   |(a19&a20&a22)|
(a20&a21)   |(a21&a22);
    pre_c22=(a15&a16&a18&a20&a22)|(a16&a17&a19&a21&a23)|(a17&a18&a20&a22)|(a18&a19&a21&a23)|(a19&a20&a22)|
(a20&a21&a23)|(a21&a22)|(a22&a23);

    pre_c23=(a23&a24);
    pre_c24=(a23&a24)           |(a24&a25);
    pre_c25=(a23&a24&a26)       |(a24&a25)           |(a25&a26);
    pre_c26=(a23&a24&a26)       |(a24&a25&a27)       |(a25&a26)       |(a26&a27);
    pre_c27=(a23&a24&a26&a28)   |(a24&a25&a27)       |(a25&a26&a28)   |(a26&a27)       |(a27&a28);
    pre_c28=(a23&a24&a26&a28)   |(a24&a25&a27&a29)   |(a25&a26&a28)   |(a26&a27&a29)   |(a27&a28)   |
(a28&a29);
    pre_c29=(a23&a24&a26&a28&a30)|(a24&a25&a27&a29)   |(a25&a26&a28&a30)|(a26&a27&a29)   |(a27&a28&a30)|
(a28&a29)   |(a29&a30);
    pre_c30=(a23&a24&a26&a28&a30)|(a24&a25&a27&a29&(!a31))|(a25&a26&a28&a30)|(a26&a27&a29&(!a31))|(a27&a28&a30)|
(a28&a29&(!a31))|(a29&a30)|(a30&(!a31));
    pre_c31=                    (a24&a25&a27&a29&(!a31))               |(a26&a27&a29&(!a31))|(a28&a29&(!
a31))                      |(a30&(!a31));


    //x3 sum(MSB negated) -> bd
    pre_s0 =     0^( 0^a0);
    pre_s1 =     0^(a0^a1);
    pre_s2 =pre_c0^(a1^a2);
    pre_s3 =pre_c1^(a2^a3);
    pre_s4 =pre_c2^(a3^a4);
    pre_s5 =pre_c3^(a4^a5);
    pre_s6 =pre_c4^(a5^a6);
    pre_s7 =pre_c5^(a6^a7);

    pre_s8 =     0^(   a7^a8);//cut->carry=0
    pre_s9 =pre_c7^(   a8^a9);
    pre_s10=pre_c8^(  a9^a10);
    pre_s11=pre_c9^( a10^a11);
    pre_s12=pre_c10^(a11^a12);
    pre_s13=pre_c11^(a12^a13);
    pre_s14=pre_c12^(a13^a14);
    pre_s15=pre_c13^(a14^a15);

    pre_s16=      0^(a15^a16);//cut->carry=0
    pre_s17=pre_c15^(a16^a17);
    pre_s18=pre_c16^(a17^a18);
    pre_s19=pre_c17^(a18^a19);
    pre_s20=pre_c18^(a19^a20);
    pre_s21=pre_c19^(a20^a21);
    pre_s22=pre_c20^(a21^a22);
    pre_s23=pre_c21^(a22^a23);

    pre_s24=      0^(a23^a24);//cut->carry=0
    pre_s25=pre_c23^(a24^a25);
    pre_s26=pre_c24^(a25^a26);
    pre_s27=pre_c25^(a26^a27);
```

```
    pre_s28=pre_c26^(a27^a28);
    pre_s29=pre_c27^(a28^a29);
    pre_s30=pre_c28^(a29^a30);
    pre_s31=pre_c29^(a30^(!a31));
    pre_s32=pre_c30^((!a31)^0  );
    pre_s33=pre_c31^(0^0    );

    //BD,bD,Bd,bd
    h0_00=(!b0)&(!b1);
    h0_01=  b0 &(!b1);
    h0_10=(!b0)&  b1;
    h0_11=  b0 &  b1;
    h1_00=(!b2)&(!b3);
    h1_01=  b2 &(!b3);
    h1_10=(!b2)&  b3;
    h1_11=  b2 &  b3;
    h2_00=(!b4)&(!b5);
    h2_01=  b4 &(!b5);
    h2_10=(!b4)&  b5;
    h2_11=  b4 &  b5;
    h3_00=(!b6)&(!b7);
    h3_01=  b6 &(!b7);
    h3_10=(!b6)&  b7;
    h3_11=  b6 &  b7;

    h4_00=(!b8 )&(!b9);
    h4_01=  b8  &(!b9);
    h4_10=(!b8 )&  b9;
    h4_11=  b8  &  b9;
    h5_00=(!b10)&(!b11);
    h5_01=  b10 &(!b11);
    h5_10=(!b10)&  b11;
    h5_11=  b10 &  b11;
    h6_00=(!b12)&(!b13);
    h6_01=  b12 &(!b13);
    h6_10=(!b12)&  b13;
    h6_11=  b12 &  b13;
    h7_00=(!b14)&(!b15);
    h7_01=  b14 &(!b15);
    h7_10=(!b14)&  b15;
    h7_11=  b14 &  b15;

    h8_00= (!b16)&(!b17);
    h8_01=   b16 &(!b17);
    h8_10= (!b16)&  b17;
    h8_11=   b16 &  b17;
    h9_00= (!b18)&(!b19);
    h9_01=   b18 &(!b19);
    h9_10= (!b18)&  b19;
    h9_11=   b18 &  b19;
    h10_00=(!b20)&(!b21);
    h10_01=  b20 &(!b21);
    h10_10=(!b20)&  b21;
    h10_11=  b20 &  b21;
    h11_00=(!b22)&(!b23);
    h11_01=  b22 &(!b23);
    h11_10=(!b22)&  b23;
    h11_11=  b22 &  b23;

    h12_00=(!b24)&(!b25);
    h12_01=  b24 &(!b25);
    h12_10=(!b24)&  b25;
    h12_11=  b24 &  b25;
    h13_00=(!b26)&(!b27);
    h13_01=  b26 &(!b27);
    h13_10=(!b26)&  b27;
    h13_11=  b26 &  b27;
    h14_00=(!b28)&(!b29);
    h14_01=  b28 &(!b29);
    h14_10=(!b28)&  b29;
    h14_11=  b28 &  b29;
    h15_00=(!b30)&(!b31);
    h15_01=  b30 &(!b31);
    h15_10=(!b30)&  b31;
    h15_11=  b30 &  b31;

    //presum
    s0_h0 = (h0_11&(pre_s0))|(h0_01&a0)|(h0_10&0 );
    s1_h0 = (h0_11&(pre_s1))|(h0_01&a1)|(h0_10&a0);
    s2_h0 = (h0_11&(pre_s2))|(h0_01&a2)|(h0_10&a1);
    s3_h0 = (h0_11&(pre_s3))|(h0_01&a3)|(h0_10&a2);
    s4_h0 = (h0_11&(pre_s4))|(h0_01&a4)|(h0_10&a3);
    s5_h0 = (h0_11&(pre_s5))|(h0_01&a5)|(h0_10&a4);
    s6_h0 = (h0_11&(pre_s6))|(h0_01&a6)|(h0_10&a5);
    s7_h0 = (h0_11&(pre_s7))|(h0_01&a7)|(h0_10&a6);
    s8_h0 = (h0_11&(pre_s8))|(h0_01&a8)|(h0_10&a7);
    s9_h0 = (h0_11&(pre_s9))|(h0_01&a9)|(h0_10&a8);
    s10_h0 = (h0_11&(pre_s10))|(h0_01&a10)|(h0_10&a9);
    s11_h0 = (h0_11&(pre_s11))|(h0_01&a11)|(h0_10&a10);
    s12_h0 = (h0_11&(pre_s12))|(h0_01&a12)|(h0_10&a11);
    s13_h0 = (h0_11&(pre_s13))|(h0_01&a13)|(h0_10&a12);
    s14_h0 = (h0_11&(pre_s14))|(h0_01&a14)|(h0_10&a13);
    s15_h0 = (h0_11&(pre_s15))|(h0_01&a15)|(h0_10&a14);
    s16_h0 = (h0_11&(pre_s16))|(h0_01&a16)|(h0_10&a15);
    s17_h0 = (h0_11&(pre_s17))|(h0_01&a17)|(h0_10&a16);
    s18_h0 = (h0_11&(pre_s18))|(h0_01&a18)|(h0_10&a17);
    s19_h0 = (h0_11&(pre_s19))|(h0_01&a19)|(h0_10&a18);
    s20_h0 = (h0_11&(pre_s20))|(h0_01&a20)|(h0_10&a19);
    s21_h0 = (h0_11&(pre_s21))|(h0_01&a21)|(h0_10&a20);
```

```
    s22_h0 = (h0_11&(pre_s22))|(h0_01&a22)|(h0_10&a21);
    s23_h0 = (h0_11&(pre_s23))|(h0_01&a23)|(h0_10&a22);
    s24_h0 = (h0_11&(pre_s24))|(h0_01&a24)|(h0_10&a23);
    s25_h0 = (h0_11&(pre_s25))|(h0_01&a25)|(h0_10&a24);
    s26_h0 = (h0_11&(pre_s26))|(h0_01&a26)|(h0_10&a25);
    s27_h0 = (h0_11&(pre_s27))|(h0_01&a27)|(h0_10&a26);
    s28_h0 = (h0_11&(pre_s28))|(h0_01&a28)|(h0_10&a27);
    s29_h0 = (h0_11&(pre_s29))|(h0_01&a29)|(h0_10&a28);
    s30_h0 = (h0_11&(pre_s30))|(h0_01&a30)|(h0_10&a29);

    s31_h0 = (h0_11&(pre_s31))|(h0_01&(!a31))|(h0_10&(!a30))      |(h0_00&1);
    s32_h0 = (h0_11&(pre_s32))|(h0_01&1 )    |(h0_10&(a30^(!a31)))|(h0_00&1);
    s33_h0 = (h0_11&(pre_s33))|(h0_01&0 )    |(h0_10&(a30&(!a31)));

    cut0_h0 = pre_c6&h0_11;
    cut1_h0 = pre_c14&h0_11;
    cut2_h0 = pre_c22&h0_11;


    s0_h1 = (h1_11&(pre_s0))|(h1_01&a0)|(h1_10&0 );
    s1_h1 = (h1_11&(pre_s1))|(h1_01&a1)|(h1_10&a0);
    s2_h1 = (h1_11&(pre_s2))|(h1_01&a2)|(h1_10&a1);
    s3_h1 = (h1_11&(pre_s3))|(h1_01&a3)|(h1_10&a2);
    s4_h1 = (h1_11&(pre_s4))|(h1_01&a4)|(h1_10&a3);
    s5_h1 = (h1_11&(pre_s5))|(h1_01&a5)|(h1_10&a4);
    s6_h1 = (h1_11&(pre_s6))|(h1_01&a6)|(h1_10&a5);
    s7_h1 = (h1_11&(pre_s7))|(h1_01&a7)|(h1_10&a6);
    s8_h1 = (h1_11&(pre_s8))|(h1_01&a8)|(h1_10&a7);
    s9_h1 = (h1_11&(pre_s9))|(h1_01&a9)|(h1_10&a8);
    s10_h1 = (h1_11&(pre_s10))|(h1_01&a10)|(h1_10&a9);
    s11_h1 = (h1_11&(pre_s11))|(h1_01&a11)|(h1_10&a10);
    s12_h1 = (h1_11&(pre_s12))|(h1_01&a12)|(h1_10&a11);
    s13_h1 = (h1_11&(pre_s13))|(h1_01&a13)|(h1_10&a12);
    s14_h1 = (h1_11&(pre_s14))|(h1_01&a14)|(h1_10&a13);
    s15_h1 = (h1_11&(pre_s15))|(h1_01&a15)|(h1_10&a14);
    s16_h1 = (h1_11&(pre_s16))|(h1_01&a16)|(h1_10&a15);
    s17_h1 = (h1_11&(pre_s17))|(h1_01&a17)|(h1_10&a16);
    s18_h1 = (h1_11&(pre_s18))|(h1_01&a18)|(h1_10&a17);
    s19_h1 = (h1_11&(pre_s19))|(h1_01&a19)|(h1_10&a18);
    s20_h1 = (h1_11&(pre_s20))|(h1_01&a20)|(h1_10&a19);
    s21_h1 = (h1_11&(pre_s21))|(h1_01&a21)|(h1_10&a20);
    s22_h1 = (h1_11&(pre_s22))|(h1_01&a22)|(h1_10&a21);
    s23_h1 = (h1_11&(pre_s23))|(h1_01&a23)|(h1_10&a22);
    s24_h1 = (h1_11&(pre_s24))|(h1_01&a24)|(h1_10&a23);
    s25_h1 = (h1_11&(pre_s25))|(h1_01&a25)|(h1_10&a24);
    s26_h1 = (h1_11&(pre_s26))|(h1_01&a26)|(h1_10&a25);
    s27_h1 = (h1_11&(pre_s27))|(h1_01&a27)|(h1_10&a26);
    s28_h1 = (h1_11&(pre_s28))|(h1_01&a28)|(h1_10&a27);
    s29_h1 = (h1_11&(pre_s29))|(h1_01&a29)|(h1_10&a28);
    s30_h1 = (h1_11&(pre_s30))|(h1_01&a30)|(h1_10&a29);

    s31_h1 = (h1_11&(pre_s31))|(h1_01&(!a31))|(h1_10&(!a30))      |(h1_00&1);
    s32_h1 = (h1_11&(pre_s32))|(h1_01&1 )    |(h1_10&(a30^(!a31)))|(h1_00&1);
    s33_h1 = (h1_11&(pre_s33))|(h1_01&0 )    |(h1_10&(a30&(!a31)));

    cut0_h1 = pre_c6&h1_11;
    cut1_h1 = pre_c14&h1_11;
    cut2_h1 = pre_c22&h1_11;


    s0_h2 = (h2_11&(pre_s0))|(h2_01&a0)|(h2_10&0 );
    s1_h2 = (h2_11&(pre_s1))|(h2_01&a1)|(h2_10&a0);
    s2_h2 = (h2_11&(pre_s2))|(h2_01&a2)|(h2_10&a1);
    s3_h2 = (h2_11&(pre_s3))|(h2_01&a3)|(h2_10&a2);
    s4_h2 = (h2_11&(pre_s4))|(h2_01&a4)|(h2_10&a3);
    s5_h2 = (h2_11&(pre_s5))|(h2_01&a5)|(h2_10&a4);
    s6_h2 = (h2_11&(pre_s6))|(h2_01&a6)|(h2_10&a5);
    s7_h2 = (h2_11&(pre_s7))|(h2_01&a7)|(h2_10&a6);
    s8_h2 = (h2_11&(pre_s8))|(h2_01&a8)|(h2_10&a7);
    s9_h2 = (h2_11&(pre_s9))|(h2_01&a9)|(h2_10&a8);
    s10_h2 = (h2_11&(pre_s10))|(h2_01&a10)|(h2_10&a9);
    s11_h2 = (h2_11&(pre_s11))|(h2_01&a11)|(h2_10&a10);
    s12_h2 = (h2_11&(pre_s12))|(h2_01&a12)|(h2_10&a11);
    s13_h2 = (h2_11&(pre_s13))|(h2_01&a13)|(h2_10&a12);
    s14_h2 = (h2_11&(pre_s14))|(h2_01&a14)|(h2_10&a13);
    s15_h2 = (h2_11&(pre_s15))|(h2_01&a15)|(h2_10&a14);
    s16_h2 = (h2_11&(pre_s16))|(h2_01&a16)|(h2_10&a15);
    s17_h2 = (h2_11&(pre_s17))|(h2_01&a17)|(h2_10&a16);
    s18_h2 = (h2_11&(pre_s18))|(h2_01&a18)|(h2_10&a17);
    s19_h2 = (h2_11&(pre_s19))|(h2_01&a19)|(h2_10&a18);
    s20_h2 = (h2_11&(pre_s20))|(h2_01&a20)|(h2_10&a19);
    s21_h2 = (h2_11&(pre_s21))|(h2_01&a21)|(h2_10&a20);
    s22_h2 = (h2_11&(pre_s22))|(h2_01&a22)|(h2_10&a21);
    s23_h2 = (h2_11&(pre_s23))|(h2_01&a23)|(h2_10&a22);
    s24_h2 = (h2_11&(pre_s24))|(h2_01&a24)|(h2_10&a23);
    s25_h2 = (h2_11&(pre_s25))|(h2_01&a25)|(h2_10&a24);
    s26_h2 = (h2_11&(pre_s26))|(h2_01&a26)|(h2_10&a25);
    s27_h2 = (h2_11&(pre_s27))|(h2_01&a27)|(h2_10&a26);
    s28_h2 = (h2_11&(pre_s28))|(h2_01&a28)|(h2_10&a27);
    s29_h2 = (h2_11&(pre_s29))|(h2_01&a29)|(h2_10&a28);
    s30_h2 = (h2_11&(pre_s30))|(h2_01&a30)|(h2_10&a29);

    s31_h2 = (h2_11&(pre_s31))|(h2_01&(!a31))|(h2_10&(!a30))      |(h2_00&1);
    s32_h2 = (h2_11&(pre_s32))|(h2_01&1 )    |(h2_10&(a30^(!a31)))|(h2_00&1);
    s33_h2 = (h2_11&(pre_s33))|(h2_01&0 )    |(h2_10&(a30&(!a31)));

    cut0_h2 = pre_c6&h2_11;
    cut1_h2 = pre_c14&h2_11;
    cut2_h2 = pre_c22&h2_11;
```

```
s0_h3 = (h3_11&(pre_s0))|(h3_01&a0)|(h3_10&0 );
s1_h3 = (h3_11&(pre_s1))|(h3_01&a1)|(h3_10&a0);
s2_h3 = (h3_11&(pre_s2))|(h3_01&a2)|(h3_10&a1);
s3_h3 = (h3_11&(pre_s3))|(h3_01&a3)|(h3_10&a2);
s4_h3 = (h3_11&(pre_s4))|(h3_01&a4)|(h3_10&a3);
s5_h3 = (h3_11&(pre_s5))|(h3_01&a5)|(h3_10&a4);
s6_h3 = (h3_11&(pre_s6))|(h3_01&a6)|(h3_10&a5);
s7_h3 = (h3_11&(pre_s7))|(h3_01&a7)|(h3_10&a6);
s8_h3 = (h3_11&(pre_s8))|(h3_01&a8)|(h3_10&a7);
s9_h3 = (h3_11&(pre_s9))|(h3_01&a9)|(h3_10&a8);
s10_h3 = (h3_11&(pre_s10))|(h3_01&a10)|(h3_10&a9);
s11_h3 = (h3_11&(pre_s11))|(h3_01&a11)|(h3_10&a10);
s12_h3 = (h3_11&(pre_s12))|(h3_01&a12)|(h3_10&a11);
s13_h3 = (h3_11&(pre_s13))|(h3_01&a13)|(h3_10&a12);
s14_h3 = (h3_11&(pre_s14))|(h3_01&a14)|(h3_10&a13);
s15_h3 = (h3_11&(pre_s15))|(h3_01&a15)|(h3_10&a14);
s16_h3 = (h3_11&(pre_s16))|(h3_01&a16)|(h3_10&a15);
s17_h3 = (h3_11&(pre_s17))|(h3_01&a17)|(h3_10&a16);
s18_h3 = (h3_11&(pre_s18))|(h3_01&a18)|(h3_10&a17);
s19_h3 = (h3_11&(pre_s19))|(h3_01&a19)|(h3_10&a18);
s20_h3 = (h3_11&(pre_s20))|(h3_01&a20)|(h3_10&a19);
s21_h3 = (h3_11&(pre_s21))|(h3_01&a21)|(h3_10&a20);
s22_h3 = (h3_11&(pre_s22))|(h3_01&a22)|(h3_10&a21);
s23_h3 = (h3_11&(pre_s23))|(h3_01&a23)|(h3_10&a22);
s24_h3 = (h3_11&(pre_s24))|(h3_01&a24)|(h3_10&a23);
s25_h3 = (h3_11&(pre_s25))|(h3_01&a25)|(h3_10&a24);
s26_h3 = (h3_11&(pre_s26))|(h3_01&a26)|(h3_10&a25);
s27_h3 = (h3_11&(pre_s27))|(h3_01&a27)|(h3_10&a26);
s28_h3 = (h3_11&(pre_s28))|(h3_01&a28)|(h3_10&a27);
s29_h3 = (h3_11&(pre_s29))|(h3_01&a29)|(h3_10&a28);
s30_h3 = (h3_11&(pre_s30))|(h3_01&a30)|(h3_10&a29);

s31_h3 = (h3_11&(pre_s31))|(h3_01&(!a31))|(h3_10&(!a30))      |(h3_00&1);
s32_h3 = (h3_11&(pre_s32))|(h3_01&1 )    |(h3_10&(a30^(!a31)))|(h3_00&1);
s33_h3 = (h3_11&(pre_s33))|(h3_01&0 )    |(h3_10&(a30&(!a31)));

cut0_h3 = pre_c6&h3_11;
cut1_h3 = pre_c14&h3_11;
cut2_h3 = pre_c22&h3_11;


s0_h4 = (h4_11&(pre_s0))|(h4_01&a0)|(h4_10&0 );
s1_h4 = (h4_11&(pre_s1))|(h4_01&a1)|(h4_10&a0);
s2_h4 = (h4_11&(pre_s2))|(h4_01&a2)|(h4_10&a1);
s3_h4 = (h4_11&(pre_s3))|(h4_01&a3)|(h4_10&a2);
s4_h4 = (h4_11&(pre_s4))|(h4_01&a4)|(h4_10&a3);
s5_h4 = (h4_11&(pre_s5))|(h4_01&a5)|(h4_10&a4);
s6_h4 = (h4_11&(pre_s6))|(h4_01&a6)|(h4_10&a5);
s7_h4 = (h4_11&(pre_s7))|(h4_01&a7)|(h4_10&a6);
s8_h4 = (h4_11&(pre_s8))|(h4_01&a8)|(h4_10&a7);
s9_h4 = (h4_11&(pre_s9))|(h4_01&a9)|(h4_10&a8);
s10_h4 = (h4_11&(pre_s10))|(h4_01&a10)|(h4_10&a9);
s11_h4 = (h4_11&(pre_s11))|(h4_01&a11)|(h4_10&a10);
s12_h4 = (h4_11&(pre_s12))|(h4_01&a12)|(h4_10&a11);
s13_h4 = (h4_11&(pre_s13))|(h4_01&a13)|(h4_10&a12);
s14_h4 = (h4_11&(pre_s14))|(h4_01&a14)|(h4_10&a13);
s15_h4 = (h4_11&(pre_s15))|(h4_01&a15)|(h4_10&a14);
s16_h4 = (h4_11&(pre_s16))|(h4_01&a16)|(h4_10&a15);
s17_h4 = (h4_11&(pre_s17))|(h4_01&a17)|(h4_10&a16);
s18_h4 = (h4_11&(pre_s18))|(h4_01&a18)|(h4_10&a17);
s19_h4 = (h4_11&(pre_s19))|(h4_01&a19)|(h4_10&a18);
s20_h4 = (h4_11&(pre_s20))|(h4_01&a20)|(h4_10&a19);
s21_h4 = (h4_11&(pre_s21))|(h4_01&a21)|(h4_10&a20);
s22_h4 = (h4_11&(pre_s22))|(h4_01&a22)|(h4_10&a21);
s23_h4 = (h4_11&(pre_s23))|(h4_01&a23)|(h4_10&a22);
s24_h4 = (h4_11&(pre_s24))|(h4_01&a24)|(h4_10&a23);
s25_h4 = (h4_11&(pre_s25))|(h4_01&a25)|(h4_10&a24);
s26_h4 = (h4_11&(pre_s26))|(h4_01&a26)|(h4_10&a25);
s27_h4 = (h4_11&(pre_s27))|(h4_01&a27)|(h4_10&a26);
s28_h4 = (h4_11&(pre_s28))|(h4_01&a28)|(h4_10&a27);
s29_h4 = (h4_11&(pre_s29))|(h4_01&a29)|(h4_10&a28);
s30_h4 = (h4_11&(pre_s30))|(h4_01&a30)|(h4_10&a29);

s31_h4 = (h4_11&(pre_s31))|(h4_01&(!a31))|(h4_10&(!a30))      |(h4_00&1);
s32_h4 = (h4_11&(pre_s32))|(h4_01&1 )    |(h4_10&(a30^(!a31)))|(h4_00&1);
s33_h4 = (h4_11&(pre_s33))|(h4_01&0 )    |(h4_10&(a30&(!a31)));

cut0_h4 = pre_c6&h4_11;
cut1_h4 = pre_c14&h4_11;
cut2_h4 = pre_c22&h4_11;


s0_h5 = (h5_11&(pre_s0))|(h5_01&a0)|(h5_10&0 );
s1_h5 = (h5_11&(pre_s1))|(h5_01&a1)|(h5_10&a0);
s2_h5 = (h5_11&(pre_s2))|(h5_01&a2)|(h5_10&a1);
s3_h5 = (h5_11&(pre_s3))|(h5_01&a3)|(h5_10&a2);
s4_h5 = (h5_11&(pre_s4))|(h5_01&a4)|(h5_10&a3);
s5_h5 = (h5_11&(pre_s5))|(h5_01&a5)|(h5_10&a4);
s6_h5 = (h5_11&(pre_s6))|(h5_01&a6)|(h5_10&a5);
s7_h5 = (h5_11&(pre_s7))|(h5_01&a7)|(h5_10&a6);
s8_h5 = (h5_11&(pre_s8))|(h5_01&a8)|(h5_10&a7);
s9_h5 = (h5_11&(pre_s9))|(h5_01&a9)|(h5_10&a8);
s10_h5 = (h5_11&(pre_s10))|(h5_01&a10)|(h5_10&a9);
s11_h5 = (h5_11&(pre_s11))|(h5_01&a11)|(h5_10&a10);
s12_h5 = (h5_11&(pre_s12))|(h5_01&a12)|(h5_10&a11);
s13_h5 = (h5_11&(pre_s13))|(h5_01&a13)|(h5_10&a12);
s14_h5 = (h5_11&(pre_s14))|(h5_01&a14)|(h5_10&a13);
```

```
s15_h5 = (h5_11&(pre_s15))|(h5_01&a15)|(h5_10&a14);
s16_h5 = (h5_11&(pre_s16))|(h5_01&a16)|(h5_10&a15);
s17_h5 = (h5_11&(pre_s17))|(h5_01&a17)|(h5_10&a16);
s18_h5 = (h5_11&(pre_s18))|(h5_01&a18)|(h5_10&a17);
s19_h5 = (h5_11&(pre_s19))|(h5_01&a19)|(h5_10&a18);
s20_h5 = (h5_11&(pre_s20))|(h5_01&a20)|(h5_10&a19);
s21_h5 = (h5_11&(pre_s21))|(h5_01&a21)|(h5_10&a20);
s22_h5 = (h5_11&(pre_s22))|(h5_01&a22)|(h5_10&a21);
s23_h5 = (h5_11&(pre_s23))|(h5_01&a23)|(h5_10&a22);
s24_h5 = (h5_11&(pre_s24))|(h5_01&a24)|(h5_10&a23);
s25_h5 = (h5_11&(pre_s25))|(h5_01&a25)|(h5_10&a24);
s26_h5 = (h5_11&(pre_s26))|(h5_01&a26)|(h5_10&a25);
s27_h5 = (h5_11&(pre_s27))|(h5_01&a27)|(h5_10&a26);
s28_h5 = (h5_11&(pre_s28))|(h5_01&a28)|(h5_10&a27);
s29_h5 = (h5_11&(pre_s29))|(h5_01&a29)|(h5_10&a28);
s30_h5 = (h5_11&(pre_s30))|(h5_01&a30)|(h5_10&a29);

s31_h5 = (h5_11&(pre_s31))|(h5_01&(!a31))|(h5_10&(!a30))      |(h5_00&1);
s32_h5 = (h5_11&(pre_s32))|(h5_01&1 )    |(h5_10&(a30^(!a31)))|(h5_00&1);
s33_h5 = (h5_11&(pre_s33))|(h5_01&0 )    |(h5_10&(a30&(!a31)));

cut0_h5 = pre_c6&h5_11;
cut1_h5 = pre_c14&h5_11;
cut2_h5 = pre_c22&h5_11;


s0_h6 = (h6_11&(pre_s0))|(h6_01&a0)|(h6_10&0 );
s1_h6 = (h6_11&(pre_s1))|(h6_01&a1)|(h6_10&a0);
s2_h6 = (h6_11&(pre_s2))|(h6_01&a2)|(h6_10&a1);
s3_h6 = (h6_11&(pre_s3))|(h6_01&a3)|(h6_10&a2);
s4_h6 = (h6_11&(pre_s4))|(h6_01&a4)|(h6_10&a3);
s5_h6 = (h6_11&(pre_s5))|(h6_01&a5)|(h6_10&a4);
s6_h6 = (h6_11&(pre_s6))|(h6_01&a6)|(h6_10&a5);
s7_h6 = (h6_11&(pre_s7))|(h6_01&a7)|(h6_10&a6);
s8_h6 = (h6_11&(pre_s8))|(h6_01&a8)|(h6_10&a7);
s9_h6 = (h6_11&(pre_s9))|(h6_01&a9)|(h6_10&a8);
s10_h6 = (h6_11&(pre_s10))|(h6_01&a10)|(h6_10&a9);
s11_h6 = (h6_11&(pre_s11))|(h6_01&a11)|(h6_10&a10);
s12_h6 = (h6_11&(pre_s12))|(h6_01&a12)|(h6_10&a11);
s13_h6 = (h6_11&(pre_s13))|(h6_01&a13)|(h6_10&a12);
s14_h6 = (h6_11&(pre_s14))|(h6_01&a14)|(h6_10&a13);
s15_h6 = (h6_11&(pre_s15))|(h6_01&a15)|(h6_10&a14);
s16_h6 = (h6_11&(pre_s16))|(h6_01&a16)|(h6_10&a15);
s17_h6 = (h6_11&(pre_s17))|(h6_01&a17)|(h6_10&a16);
s18_h6 = (h6_11&(pre_s18))|(h6_01&a18)|(h6_10&a17);
s19_h6 = (h6_11&(pre_s19))|(h6_01&a19)|(h6_10&a18);
s20_h6 = (h6_11&(pre_s20))|(h6_01&a20)|(h6_10&a19);
s21_h6 = (h6_11&(pre_s21))|(h6_01&a21)|(h6_10&a20);
s22_h6 = (h6_11&(pre_s22))|(h6_01&a22)|(h6_10&a21);
s23_h6 = (h6_11&(pre_s23))|(h6_01&a23)|(h6_10&a22);
s24_h6 = (h6_11&(pre_s24))|(h6_01&a24)|(h6_10&a23);
s25_h6 = (h6_11&(pre_s25))|(h6_01&a25)|(h6_10&a24);
s26_h6 = (h6_11&(pre_s26))|(h6_01&a26)|(h6_10&a25);
s27_h6 = (h6_11&(pre_s27))|(h6_01&a27)|(h6_10&a26);
s28_h6 = (h6_11&(pre_s28))|(h6_01&a28)|(h6_10&a27);
s29_h6 = (h6_11&(pre_s29))|(h6_01&a29)|(h6_10&a28);
s30_h6 = (h6_11&(pre_s30))|(h6_01&a30)|(h6_10&a29);

s31_h6 = (h6_11&(pre_s31))|(h6_01&(!a31))|(h6_10&(!a30))      |(h6_00&1);
s32_h6 = (h6_11&(pre_s32))|(h6_01&1 )    |(h6_10&(a30^(!a31)))|(h6_00&1);
s33_h6 = (h6_11&(pre_s33))|(h6_01&0 )    |(h6_10&(a30&(!a31)));

cut0_h6 = pre_c6&h6_11;
cut1_h6 = pre_c14&h6_11;
cut2_h6 = pre_c22&h6_11;


s0_h7 = (h7_11&(pre_s0))|(h7_01&a0)|(h7_10&0 );
s1_h7 = (h7_11&(pre_s1))|(h7_01&a1)|(h7_10&a0);
s2_h7 = (h7_11&(pre_s2))|(h7_01&a2)|(h7_10&a1);
s3_h7 = (h7_11&(pre_s3))|(h7_01&a3)|(h7_10&a2);
s4_h7 = (h7_11&(pre_s4))|(h7_01&a4)|(h7_10&a3);
s5_h7 = (h7_11&(pre_s5))|(h7_01&a5)|(h7_10&a4);
s6_h7 = (h7_11&(pre_s6))|(h7_01&a6)|(h7_10&a5);
s7_h7 = (h7_11&(pre_s7))|(h7_01&a7)|(h7_10&a6);
s8_h7 = (h7_11&(pre_s8))|(h7_01&a8)|(h7_10&a7);
s9_h7 = (h7_11&(pre_s9))|(h7_01&a9)|(h7_10&a8);
s10_h7 = (h7_11&(pre_s10))|(h7_01&a10)|(h7_10&a9);
s11_h7 = (h7_11&(pre_s11))|(h7_01&a11)|(h7_10&a10);
s12_h7 = (h7_11&(pre_s12))|(h7_01&a12)|(h7_10&a11);
s13_h7 = (h7_11&(pre_s13))|(h7_01&a13)|(h7_10&a12);
s14_h7 = (h7_11&(pre_s14))|(h7_01&a14)|(h7_10&a13);
s15_h7 = (h7_11&(pre_s15))|(h7_01&a15)|(h7_10&a14);
s16_h7 = (h7_11&(pre_s16))|(h7_01&a16)|(h7_10&a15);
s17_h7 = (h7_11&(pre_s17))|(h7_01&a17)|(h7_10&a16);
s18_h7 = (h7_11&(pre_s18))|(h7_01&a18)|(h7_10&a17);
s19_h7 = (h7_11&(pre_s19))|(h7_01&a19)|(h7_10&a18);
s20_h7 = (h7_11&(pre_s20))|(h7_01&a20)|(h7_10&a19);
s21_h7 = (h7_11&(pre_s21))|(h7_01&a21)|(h7_10&a20);
s22_h7 = (h7_11&(pre_s22))|(h7_01&a22)|(h7_10&a21);
s23_h7 = (h7_11&(pre_s23))|(h7_01&a23)|(h7_10&a22);
s24_h7 = (h7_11&(pre_s24))|(h7_01&a24)|(h7_10&a23);
s25_h7 = (h7_11&(pre_s25))|(h7_01&a25)|(h7_10&a24);
s26_h7 = (h7_11&(pre_s26))|(h7_01&a26)|(h7_10&a25);
s27_h7 = (h7_11&(pre_s27))|(h7_01&a27)|(h7_10&a26);
s28_h7 = (h7_11&(pre_s28))|(h7_01&a28)|(h7_10&a27);
s29_h7 = (h7_11&(pre_s29))|(h7_01&a29)|(h7_10&a28);
s30_h7 = (h7_11&(pre_s30))|(h7_01&a30)|(h7_10&a29);
```

```
    s31_h7 = (h7_11&(pre_s31))|(h7_01&(!a31))|(h7_10&(!a30))      |(h7_00&1);
    s32_h7 = (h7_11&(pre_s32))|(h7_01&1 )    |(h7_10&(a30^(!a31)))|(h7_00&1);
    s33_h7 = (h7_11&(pre_s33))|(h7_01&0 )    |(h7_10&(a30&(!a31)));

    cut0_h7 = pre_c6&h7_11;
    cut1_h7 = pre_c14&h7_11;
    cut2_h7 = pre_c22&h7_11;


    s0_h8 = (h8_11&(pre_s0))|(h8_01&a0)|(h8_10&0 );
    s1_h8 = (h8_11&(pre_s1))|(h8_01&a1)|(h8_10&a0);
    s2_h8 = (h8_11&(pre_s2))|(h8_01&a2)|(h8_10&a1);
    s3_h8 = (h8_11&(pre_s3))|(h8_01&a3)|(h8_10&a2);
    s4_h8 = (h8_11&(pre_s4))|(h8_01&a4)|(h8_10&a3);
    s5_h8 = (h8_11&(pre_s5))|(h8_01&a5)|(h8_10&a4);
    s6_h8 = (h8_11&(pre_s6))|(h8_01&a6)|(h8_10&a5);
    s7_h8 = (h8_11&(pre_s7))|(h8_01&a7)|(h8_10&a6);
    s8_h8 = (h8_11&(pre_s8))|(h8_01&a8)|(h8_10&a7);
    s9_h8 = (h8_11&(pre_s9))|(h8_01&a9)|(h8_10&a8);
    s10_h8 = (h8_11&(pre_s10))|(h8_01&a10)|(h8_10&a9);
    s11_h8 = (h8_11&(pre_s11))|(h8_01&a11)|(h8_10&a10);
    s12_h8 = (h8_11&(pre_s12))|(h8_01&a12)|(h8_10&a11);
    s13_h8 = (h8_11&(pre_s13))|(h8_01&a13)|(h8_10&a12);
    s14_h8 = (h8_11&(pre_s14))|(h8_01&a14)|(h8_10&a13);
    s15_h8 = (h8_11&(pre_s15))|(h8_01&a15)|(h8_10&a14);
    s16_h8 = (h8_11&(pre_s16))|(h8_01&a16)|(h8_10&a15);
    s17_h8 = (h8_11&(pre_s17))|(h8_01&a17)|(h8_10&a16);
    s18_h8 = (h8_11&(pre_s18))|(h8_01&a18)|(h8_10&a17);
    s19_h8 = (h8_11&(pre_s19))|(h8_01&a19)|(h8_10&a18);
    s20_h8 = (h8_11&(pre_s20))|(h8_01&a20)|(h8_10&a19);
    s21_h8 = (h8_11&(pre_s21))|(h8_01&a21)|(h8_10&a20);
    s22_h8 = (h8_11&(pre_s22))|(h8_01&a22)|(h8_10&a21);
    s23_h8 = (h8_11&(pre_s23))|(h8_01&a23)|(h8_10&a22);
    s24_h8 = (h8_11&(pre_s24))|(h8_01&a24)|(h8_10&a23);
    s25_h8 = (h8_11&(pre_s25))|(h8_01&a25)|(h8_10&a24);
    s26_h8 = (h8_11&(pre_s26))|(h8_01&a26)|(h8_10&a25);
    s27_h8 = (h8_11&(pre_s27))|(h8_01&a27)|(h8_10&a26);
    s28_h8 = (h8_11&(pre_s28))|(h8_01&a28)|(h8_10&a27);
    s29_h8 = (h8_11&(pre_s29))|(h8_01&a29)|(h8_10&a28);
    s30_h8 = (h8_11&(pre_s30))|(h8_01&a30)|(h8_10&a29);

    s31_h8 = (h8_11&(pre_s31))|(h8_01&(!a31))|(h8_10&(!a30))      |(h8_00&1);
    s32_h8 = (h8_11&(pre_s32))|(h8_01&1 )    |(h8_10&(a30^(!a31)))|(h8_00&1);
    s33_h8 = (h8_11&(pre_s33))|(h8_01&0 )    |(h8_10&(a30&(!a31)));

    cut0_h8 = pre_c6&h8_11;
    cut1_h8 = pre_c14&h8_11;
    cut2_h8 = pre_c22&h8_11;


    s0_h9 = (h9_11&(pre_s0))|(h9_01&a0)|(h9_10&0 );
    s1_h9 = (h9_11&(pre_s1))|(h9_01&a1)|(h9_10&a0);
    s2_h9 = (h9_11&(pre_s2))|(h9_01&a2)|(h9_10&a1);
    s3_h9 = (h9_11&(pre_s3))|(h9_01&a3)|(h9_10&a2);
    s4_h9 = (h9_11&(pre_s4))|(h9_01&a4)|(h9_10&a3);
    s5_h9 = (h9_11&(pre_s5))|(h9_01&a5)|(h9_10&a4);
    s6_h9 = (h9_11&(pre_s6))|(h9_01&a6)|(h9_10&a5);
    s7_h9 = (h9_11&(pre_s7))|(h9_01&a7)|(h9_10&a6);
    s8_h9 = (h9_11&(pre_s8))|(h9_01&a8)|(h9_10&a7);
    s9_h9 = (h9_11&(pre_s9))|(h9_01&a9)|(h9_10&a8);
    s10_h9 = (h9_11&(pre_s10))|(h9_01&a10)|(h9_10&a9);
    s11_h9 = (h9_11&(pre_s11))|(h9_01&a11)|(h9_10&a10);
    s12_h9 = (h9_11&(pre_s12))|(h9_01&a12)|(h9_10&a11);
    s13_h9 = (h9_11&(pre_s13))|(h9_01&a13)|(h9_10&a12);
    s14_h9 = (h9_11&(pre_s14))|(h9_01&a14)|(h9_10&a13);
    s15_h9 = (h9_11&(pre_s15))|(h9_01&a15)|(h9_10&a14);
    s16_h9 = (h9_11&(pre_s16))|(h9_01&a16)|(h9_10&a15);
    s17_h9 = (h9_11&(pre_s17))|(h9_01&a17)|(h9_10&a16);
    s18_h9 = (h9_11&(pre_s18))|(h9_01&a18)|(h9_10&a17);
    s19_h9 = (h9_11&(pre_s19))|(h9_01&a19)|(h9_10&a18);
    s20_h9 = (h9_11&(pre_s20))|(h9_01&a20)|(h9_10&a19);
    s21_h9 = (h9_11&(pre_s21))|(h9_01&a21)|(h9_10&a20);
    s22_h9 = (h9_11&(pre_s22))|(h9_01&a22)|(h9_10&a21);
    s23_h9 = (h9_11&(pre_s23))|(h9_01&a23)|(h9_10&a22);
    s24_h9 = (h9_11&(pre_s24))|(h9_01&a24)|(h9_10&a23);
    s25_h9 = (h9_11&(pre_s25))|(h9_01&a25)|(h9_10&a24);
    s26_h9 = (h9_11&(pre_s26))|(h9_01&a26)|(h9_10&a25);
    s27_h9 = (h9_11&(pre_s27))|(h9_01&a27)|(h9_10&a26);
    s28_h9 = (h9_11&(pre_s28))|(h9_01&a28)|(h9_10&a27);
    s29_h9 = (h9_11&(pre_s29))|(h9_01&a29)|(h9_10&a28);
    s30_h9 = (h9_11&(pre_s30))|(h9_01&a30)|(h9_10&a29);

    s31_h9 = (h9_11&(pre_s31))|(h9_01&(!a31))|(h9_10&(!a30))      |(h9_00&1);
    s32_h9 = (h9_11&(pre_s32))|(h9_01&1 )    |(h9_10&(a30^(!a31)))|(h9_00&1);
    s33_h9 = (h9_11&(pre_s33))|(h9_01&0 )    |(h9_10&(a30&(!a31)));

    cut0_h9 = pre_c6&h9_11;
    cut1_h9 = pre_c14&h9_11;
    cut2_h9 = pre_c22&h9_11;


    s0_h10 = (h10_11&(pre_s0))|(h10_01&a0)|(h10_10&0 );
    s1_h10 = (h10_11&(pre_s1))|(h10_01&a1)|(h10_10&a0);
    s2_h10 = (h10_11&(pre_s2))|(h10_01&a2)|(h10_10&a1);
    s3_h10 = (h10_11&(pre_s3))|(h10_01&a3)|(h10_10&a2);
    s4_h10 = (h10_11&(pre_s4))|(h10_01&a4)|(h10_10&a3);
    s5_h10 = (h10_11&(pre_s5))|(h10_01&a5)|(h10_10&a4);
    s6_h10 = (h10_11&(pre_s6))|(h10_01&a6)|(h10_10&a5);
    s7_h10 = (h10_11&(pre_s7))|(h10_01&a7)|(h10_10&a6);
```

```
    s8_h10 = (h10_11&(pre_s8))|(h10_01&a8)|(h10_10&a7);
    s9_h10 = (h10_11&(pre_s9))|(h10_01&a9)|(h10_10&a8);
    s10_h10 = (h10_11&(pre_s10))|(h10_01&a10)|(h10_10&a9);
    s11_h10 = (h10_11&(pre_s11))|(h10_01&a11)|(h10_10&a10);
    s12_h10 = (h10_11&(pre_s12))|(h10_01&a12)|(h10_10&a11);
    s13_h10 = (h10_11&(pre_s13))|(h10_01&a13)|(h10_10&a12);
    s14_h10 = (h10_11&(pre_s14))|(h10_01&a14)|(h10_10&a13);
    s15_h10 = (h10_11&(pre_s15))|(h10_01&a15)|(h10_10&a14);
    s16_h10 = (h10_11&(pre_s16))|(h10_01&a16)|(h10_10&a15);
    s17_h10 = (h10_11&(pre_s17))|(h10_01&a17)|(h10_10&a16);
    s18_h10 = (h10_11&(pre_s18))|(h10_01&a18)|(h10_10&a17);
    s19_h10 = (h10_11&(pre_s19))|(h10_01&a19)|(h10_10&a18);
    s20_h10 = (h10_11&(pre_s20))|(h10_01&a20)|(h10_10&a19);
    s21_h10 = (h10_11&(pre_s21))|(h10_01&a21)|(h10_10&a20);
    s22_h10 = (h10_11&(pre_s22))|(h10_01&a22)|(h10_10&a21);
    s23_h10 = (h10_11&(pre_s23))|(h10_01&a23)|(h10_10&a22);
    s24_h10 = (h10_11&(pre_s24))|(h10_01&a24)|(h10_10&a23);
    s25_h10 = (h10_11&(pre_s25))|(h10_01&a25)|(h10_10&a24);
    s26_h10 = (h10_11&(pre_s26))|(h10_01&a26)|(h10_10&a25);
    s27_h10 = (h10_11&(pre_s27))|(h10_01&a27)|(h10_10&a26);
    s28_h10 = (h10_11&(pre_s28))|(h10_01&a28)|(h10_10&a27);
    s29_h10 = (h10_11&(pre_s29))|(h10_01&a29)|(h10_10&a28);
    s30_h10 = (h10_11&(pre_s30))|(h10_01&a30)|(h10_10&a29);

    s31_h10 = (h10_11&(pre_s31))|(h10_01&(!a31))|(h10_10&(!a30))      |(h10_00&1);
    s32_h10 = (h10_11&(pre_s32))|(h10_01&1 )    |(h10_10&(a30^(!a31)))|(h10_00&1);
    s33_h10 = (h10_11&(pre_s33))|(h10_01&0 )    |(h10_10&(a30&(!a31)));

    cut0_h10 = pre_c6&h10_11;
    cut1_h10 = pre_c14&h10_11;
    cut2_h10 = pre_c22&h10_11;


    s0_h11 = (h11_11&(pre_s0))|(h11_01&a0)|(h11_10&0 );
    s1_h11 = (h11_11&(pre_s1))|(h11_01&a1)|(h11_10&a0);
    s2_h11 = (h11_11&(pre_s2))|(h11_01&a2)|(h11_10&a1);
    s3_h11 = (h11_11&(pre_s3))|(h11_01&a3)|(h11_10&a2);
    s4_h11 = (h11_11&(pre_s4))|(h11_01&a4)|(h11_10&a3);
    s5_h11 = (h11_11&(pre_s5))|(h11_01&a5)|(h11_10&a4);
    s6_h11 = (h11_11&(pre_s6))|(h11_01&a6)|(h11_10&a5);
    s7_h11 = (h11_11&(pre_s7))|(h11_01&a7)|(h11_10&a6);
    s8_h11 = (h11_11&(pre_s8))|(h11_01&a8)|(h11_10&a7);
    s9_h11 = (h11_11&(pre_s9))|(h11_01&a9)|(h11_10&a8);
    s10_h11 = (h11_11&(pre_s10))|(h11_01&a10)|(h11_10&a9);
    s11_h11 = (h11_11&(pre_s11))|(h11_01&a11)|(h11_10&a10);
    s12_h11 = (h11_11&(pre_s12))|(h11_01&a12)|(h11_10&a11);
    s13_h11 = (h11_11&(pre_s13))|(h11_01&a13)|(h11_10&a12);
    s14_h11 = (h11_11&(pre_s14))|(h11_01&a14)|(h11_10&a13);
    s15_h11 = (h11_11&(pre_s15))|(h11_01&a15)|(h11_10&a14);
    s16_h11 = (h11_11&(pre_s16))|(h11_01&a16)|(h11_10&a15);
    s17_h11 = (h11_11&(pre_s17))|(h11_01&a17)|(h11_10&a16);
    s18_h11 = (h11_11&(pre_s18))|(h11_01&a18)|(h11_10&a17);
    s19_h11 = (h11_11&(pre_s19))|(h11_01&a19)|(h11_10&a18);
    s20_h11 = (h11_11&(pre_s20))|(h11_01&a20)|(h11_10&a19);
    s21_h11 = (h11_11&(pre_s21))|(h11_01&a21)|(h11_10&a20);
    s22_h11 = (h11_11&(pre_s22))|(h11_01&a22)|(h11_10&a21);
    s23_h11 = (h11_11&(pre_s23))|(h11_01&a23)|(h11_10&a22);
    s24_h11 = (h11_11&(pre_s24))|(h11_01&a24)|(h11_10&a23);
    s25_h11 = (h11_11&(pre_s25))|(h11_01&a25)|(h11_10&a24);
    s26_h11 = (h11_11&(pre_s26))|(h11_01&a26)|(h11_10&a25);
    s27_h11 = (h11_11&(pre_s27))|(h11_01&a27)|(h11_10&a26);
    s28_h11 = (h11_11&(pre_s28))|(h11_01&a28)|(h11_10&a27);
    s29_h11 = (h11_11&(pre_s29))|(h11_01&a29)|(h11_10&a28);
    s30_h11 = (h11_11&(pre_s30))|(h11_01&a30)|(h11_10&a29);

    s31_h11 = (h11_11&(pre_s31))|(h11_01&(!a31))|(h11_10&(!a30))      |(h11_00&1);
    s32_h11 = (h11_11&(pre_s32))|(h11_01&1 )    |(h11_10&(a30^(!a31)))|(h11_00&1);
    s33_h11 = (h11_11&(pre_s33))|(h11_01&0 )    |(h11_10&(a30&(!a31)));

    cut0_h11 = pre_c6&h11_11;
    cut1_h11 = pre_c14&h11_11;
    cut2_h11 = pre_c22&h11_11;


    s0_h12 = (h12_11&(pre_s0))|(h12_01&a0)|(h12_10&0 );
    s1_h12 = (h12_11&(pre_s1))|(h12_01&a1)|(h12_10&a0);
    s2_h12 = (h12_11&(pre_s2))|(h12_01&a2)|(h12_10&a1);
    s3_h12 = (h12_11&(pre_s3))|(h12_01&a3)|(h12_10&a2);
    s4_h12 = (h12_11&(pre_s4))|(h12_01&a4)|(h12_10&a3);
    s5_h12 = (h12_11&(pre_s5))|(h12_01&a5)|(h12_10&a4);
    s6_h12 = (h12_11&(pre_s6))|(h12_01&a6)|(h12_10&a5);
    s7_h12 = (h12_11&(pre_s7))|(h12_01&a7)|(h12_10&a6);
    s8_h12 = (h12_11&(pre_s8))|(h12_01&a8)|(h12_10&a7);
    s9_h12 = (h12_11&(pre_s9))|(h12_01&a9)|(h12_10&a8);
    s10_h12 = (h12_11&(pre_s10))|(h12_01&a10)|(h12_10&a9);
    s11_h12 = (h12_11&(pre_s11))|(h12_01&a11)|(h12_10&a10);
    s12_h12 = (h12_11&(pre_s12))|(h12_01&a12)|(h12_10&a11);
    s13_h12 = (h12_11&(pre_s13))|(h12_01&a13)|(h12_10&a12);
    s14_h12 = (h12_11&(pre_s14))|(h12_01&a14)|(h12_10&a13);
    s15_h12 = (h12_11&(pre_s15))|(h12_01&a15)|(h12_10&a14);
    s16_h12 = (h12_11&(pre_s16))|(h12_01&a16)|(h12_10&a15);
    s17_h12 = (h12_11&(pre_s17))|(h12_01&a17)|(h12_10&a16);
    s18_h12 = (h12_11&(pre_s18))|(h12_01&a18)|(h12_10&a17);
    s19_h12 = (h12_11&(pre_s19))|(h12_01&a19)|(h12_10&a18);
    s20_h12 = (h12_11&(pre_s20))|(h12_01&a20)|(h12_10&a19);
    s21_h12 = (h12_11&(pre_s21))|(h12_01&a21)|(h12_10&a20);
    s22_h12 = (h12_11&(pre_s22))|(h12_01&a22)|(h12_10&a21);
    s23_h12 = (h12_11&(pre_s23))|(h12_01&a23)|(h12_10&a22);
    s24_h12 = (h12_11&(pre_s24))|(h12_01&a24)|(h12_10&a23);
```

```
s25_h12 = (h12_11&(pre_s25))|(h12_01&a25)|(h12_10&a24);
s26_h12 = (h12_11&(pre_s26))|(h12_01&a26)|(h12_10&a25);
s27_h12 = (h12_11&(pre_s27))|(h12_01&a27)|(h12_10&a26);
s28_h12 = (h12_11&(pre_s28))|(h12_01&a28)|(h12_10&a27);
s29_h12 = (h12_11&(pre_s29))|(h12_01&a29)|(h12_10&a28);
s30_h12 = (h12_11&(pre_s30))|(h12_01&a30)|(h12_10&a29);

s31_h12 = (h12_11&(pre_s31))|(h12_01&(!a31))|(h12_10&(!a30))      |(h12_00&1);
s32_h12 = (h12_11&(pre_s32))|(h12_01&1 )    |(h12_10&(a30^(!a31)))|(h12_00&1);
s33_h12 = (h12_11&(pre_s33))|(h12_01&0 )    |(h12_10&(a30&(!a31)));

cut0_h12 = pre_c6&h12_11;
cut1_h12 = pre_c14&h12_11;
cut2_h12 = pre_c22&h12_11;


s0_h13 = (h13_11&(pre_s0))|(h13_01&a0)|(h13_10&0 );
s1_h13 = (h13_11&(pre_s1))|(h13_01&a1)|(h13_10&a0);
s2_h13 = (h13_11&(pre_s2))|(h13_01&a2)|(h13_10&a1);
s3_h13 = (h13_11&(pre_s3))|(h13_01&a3)|(h13_10&a2);
s4_h13 = (h13_11&(pre_s4))|(h13_01&a4)|(h13_10&a3);
s5_h13 = (h13_11&(pre_s5))|(h13_01&a5)|(h13_10&a4);
s6_h13 = (h13_11&(pre_s6))|(h13_01&a6)|(h13_10&a5);
s7_h13 = (h13_11&(pre_s7))|(h13_01&a7)|(h13_10&a6);
s8_h13 = (h13_11&(pre_s8))|(h13_01&a8)|(h13_10&a7);
s9_h13 = (h13_11&(pre_s9))|(h13_01&a9)|(h13_10&a8);
s10_h13 = (h13_11&(pre_s10))|(h13_01&a10)|(h13_10&a9);
s11_h13 = (h13_11&(pre_s11))|(h13_01&a11)|(h13_10&a10);
s12_h13 = (h13_11&(pre_s12))|(h13_01&a12)|(h13_10&a11);
s13_h13 = (h13_11&(pre_s13))|(h13_01&a13)|(h13_10&a12);
s14_h13 = (h13_11&(pre_s14))|(h13_01&a14)|(h13_10&a13);
s15_h13 = (h13_11&(pre_s15))|(h13_01&a15)|(h13_10&a14);
s16_h13 = (h13_11&(pre_s16))|(h13_01&a16)|(h13_10&a15);
s17_h13 = (h13_11&(pre_s17))|(h13_01&a17)|(h13_10&a16);
s18_h13 = (h13_11&(pre_s18))|(h13_01&a18)|(h13_10&a17);
s19_h13 = (h13_11&(pre_s19))|(h13_01&a19)|(h13_10&a18);
s20_h13 = (h13_11&(pre_s20))|(h13_01&a20)|(h13_10&a19);
s21_h13 = (h13_11&(pre_s21))|(h13_01&a21)|(h13_10&a20);
s22_h13 = (h13_11&(pre_s22))|(h13_01&a22)|(h13_10&a21);
s23_h13 = (h13_11&(pre_s23))|(h13_01&a23)|(h13_10&a22);
s24_h13 = (h13_11&(pre_s24))|(h13_01&a24)|(h13_10&a23);
s25_h13 = (h13_11&(pre_s25))|(h13_01&a25)|(h13_10&a24);
s26_h13 = (h13_11&(pre_s26))|(h13_01&a26)|(h13_10&a25);
s27_h13 = (h13_11&(pre_s27))|(h13_01&a27)|(h13_10&a26);
s28_h13 = (h13_11&(pre_s28))|(h13_01&a28)|(h13_10&a27);
s29_h13 = (h13_11&(pre_s29))|(h13_01&a29)|(h13_10&a28);
s30_h13 = (h13_11&(pre_s30))|(h13_01&a30)|(h13_10&a29);

s31_h13 = (h13_11&(pre_s31))|(h13_01&(!a31))|(h13_10&(!a30))      |(h13_00&1);
s32_h13 = (h13_11&(pre_s32))|(h13_01&1 )    |(h13_10&(a30^(!a31)))|(h13_00&1);
s33_h13 = (h13_11&(pre_s33))|(h13_01&0 )    |(h13_10&(a30&(!a31)));

cut0_h13 = pre_c6&h13_11;
cut1_h13 = pre_c14&h13_11;
cut2_h13 = pre_c22&h13_11;


s0_h14 = (h14_11&(pre_s0))|(h14_01&a0)|(h14_10&0 );
s1_h14 = (h14_11&(pre_s1))|(h14_01&a1)|(h14_10&a0);
s2_h14 = (h14_11&(pre_s2))|(h14_01&a2)|(h14_10&a1);
s3_h14 = (h14_11&(pre_s3))|(h14_01&a3)|(h14_10&a2);
s4_h14 = (h14_11&(pre_s4))|(h14_01&a4)|(h14_10&a3);
s5_h14 = (h14_11&(pre_s5))|(h14_01&a5)|(h14_10&a4);
s6_h14 = (h14_11&(pre_s6))|(h14_01&a6)|(h14_10&a5);
s7_h14 = (h14_11&(pre_s7))|(h14_01&a7)|(h14_10&a6);
s8_h14 = (h14_11&(pre_s8))|(h14_01&a8)|(h14_10&a7);
s9_h14 = (h14_11&(pre_s9))|(h14_01&a9)|(h14_10&a8);
s10_h14 = (h14_11&(pre_s10))|(h14_01&a10)|(h14_10&a9);
s11_h14 = (h14_11&(pre_s11))|(h14_01&a11)|(h14_10&a10);
s12_h14 = (h14_11&(pre_s12))|(h14_01&a12)|(h14_10&a11);
s13_h14 = (h14_11&(pre_s13))|(h14_01&a13)|(h14_10&a12);
s14_h14 = (h14_11&(pre_s14))|(h14_01&a14)|(h14_10&a13);
s15_h14 = (h14_11&(pre_s15))|(h14_01&a15)|(h14_10&a14);
s16_h14 = (h14_11&(pre_s16))|(h14_01&a16)|(h14_10&a15);
s17_h14 = (h14_11&(pre_s17))|(h14_01&a17)|(h14_10&a16);
s18_h14 = (h14_11&(pre_s18))|(h14_01&a18)|(h14_10&a17);
s19_h14 = (h14_11&(pre_s19))|(h14_01&a19)|(h14_10&a18);
s20_h14 = (h14_11&(pre_s20))|(h14_01&a20)|(h14_10&a19);
s21_h14 = (h14_11&(pre_s21))|(h14_01&a21)|(h14_10&a20);
s22_h14 = (h14_11&(pre_s22))|(h14_01&a22)|(h14_10&a21);
s23_h14 = (h14_11&(pre_s23))|(h14_01&a23)|(h14_10&a22);
s24_h14 = (h14_11&(pre_s24))|(h14_01&a24)|(h14_10&a23);
s25_h14 = (h14_11&(pre_s25))|(h14_01&a25)|(h14_10&a24);
s26_h14 = (h14_11&(pre_s26))|(h14_01&a26)|(h14_10&a25);
s27_h14 = (h14_11&(pre_s27))|(h14_01&a27)|(h14_10&a26);
s28_h14 = (h14_11&(pre_s28))|(h14_01&a28)|(h14_10&a27);
s29_h14 = (h14_11&(pre_s29))|(h14_01&a29)|(h14_10&a28);
s30_h14 = (h14_11&(pre_s30))|(h14_01&a30)|(h14_10&a29);

s31_h14 = (h14_11&(pre_s31))|(h14_01&(!a31))|(h14_10&(!a30))      |(h14_00&1);
s32_h14 = (h14_11&(pre_s32))|(h14_01&1 )    |(h14_10&(a30^(!a31)))|(h14_00&1);
s33_h14 = (h14_11&(pre_s33))|(h14_01&0 )    |(h14_10&(a30&(!a31)));

cut0_h14 = pre_c6&h14_11;
cut1_h14 = pre_c14&h14_11;
cut2_h14 = pre_c22&h14_11;


s0_h15  = (h15_11&(!a0)) |(h15_01&a0);
```

```
    s1_h15  = (h15_11&(!a1)) |(h15_01&a1) |(h15_10&(!a0));
    s2_h15  = (h15_11&(!a2)) |(h15_01&a2) |(h15_10&(!a1));
    s3_h15  = (h15_11&(!a3)) |(h15_01&a3) |(h15_10&(!a2));
    s4_h15  = (h15_11&(!a4)) |(h15_01&a4) |(h15_10&(!a3));
    s5_h15  = (h15_11&(!a5)) |(h15_01&a5) |(h15_10&(!a4));
    s6_h15  = (h15_11&(!a6)) |(h15_01&a6) |(h15_10&(!a5));
    s7_h15  = (h15_11&(!a7)) |(h15_01&a7) |(h15_10&(!a6));
    s8_h15  = (h15_11&(!a8)) |(h15_01&a8) |(h15_10&(!a7));
    s9_h15  = (h15_11&(!a9)) |(h15_01&a9) |(h15_10&(!a8));
    s10_h15 = (h15_11&(!a10))|(h15_01&a10)|(h15_10&(!a9));
    s11_h15 = (h15_11&(!a11))|(h15_01&a11)|(h15_10&(!a10));
    s12_h15 = (h15_11&(!a12))|(h15_01&a12)|(h15_10&(!a11));
    s13_h15 = (h15_11&(!a13))|(h15_01&a13)|(h15_10&(!a12));
    s14_h15 = (h15_11&(!a14))|(h15_01&a14)|(h15_10&(!a13));
    s15_h15 = (h15_11&(!a15))|(h15_01&a15)|(h15_10&(!a14));
    s16_h15 = (h15_11&(!a16))|(h15_01&a16)|(h15_10&(!a15));
    s17_h15 = (h15_11&(!a17))|(h15_01&a17)|(h15_10&(!a16));
    s18_h15 = (h15_11&(!a18))|(h15_01&a18)|(h15_10&(!a17));
    s19_h15 = (h15_11&(!a19))|(h15_01&a19)|(h15_10&(!a18));
    s20_h15 = (h15_11&(!a20))|(h15_01&a20)|(h15_10&(!a19));
    s21_h15 = (h15_11&(!a21))|(h15_01&a21)|(h15_10&(!a20));
    s22_h15 = (h15_11&(!a22))|(h15_01&a22)|(h15_10&(!a21));
    s23_h15 = (h15_11&(!a23))|(h15_01&a23)|(h15_10&(!a22));
    s24_h15 = (h15_11&(!a24))|(h15_01&a24)|(h15_10&(!a23));
    s25_h15 = (h15_11&(!a25))|(h15_01&a25)|(h15_10&(!a24));
    s26_h15 = (h15_11&(!a26))|(h15_01&a26)|(h15_10&(!a25));
    s27_h15 = (h15_11&(!a27))|(h15_01&a27)|(h15_10&(!a26));
    s28_h15 = (h15_11&(!a28))|(h15_01&a28)|(h15_10&(!a27));
    s29_h15 = (h15_11&(!a29))|(h15_01&a29)|(h15_10&(!a28));
    s30_h15 = (h15_11&(!a30))|(h15_01&a30)|(h15_10&(!a29));

    s31_h15 = (h15_11&a31)   |(h15_01&(!a31))|(h15_10&a30)            |(h15_00&1);
    s32_h15 = (h15_11&1)     |(h15_01&1)     |(h15_10&((!a30)^a31))   |(h15_00&1);
    s33_h15 = (h15_11&1)     |(h15_01&1)     |(h15_10&(!((!a30)&a31)))|(h15_00&1);

    ext0_h15 = h15_11;
    ext1_h15 = 1;
    ext2_h15 = h15_10;

    sum0 = pow33*s33_h0+pow32*s32_h0+pow31*s31_h0+pow30*s30_h0

+pow29*s29_h0+pow28*s28_h0+pow27*s27_h0+pow26*s26_h0+pow25*s25_h0+pow24*s24_h0+pow23*s23_h0+pow22*s22_h0+pow21*s2
1_h0+pow20*s20_h0

+pow19*s19_h0+pow18*s18_h0+pow17*s17_h0+pow16*s16_h0+pow15*s15_h0+pow14*s14_h0+pow13*s13_h0+pow12*s12_h0+pow11*s1
1_h0+pow10*s10_h0

+pow09*s9_h0+pow08*s8_h0+pow07*s7_h0+pow06*s6_h0+pow05*s5_h0+pow04*s4_h0+pow03*s3_h0+pow02*s2_h0+pow01*s1_h0+pow0
0*s0_h0
      +pow08*cut0_h0+pow16*cut1_h0+pow24*cut2_h0;
    sum1 = pow33*s33_h1+pow32*s32_h1+pow31*s31_h1+pow30*s30_h1

+pow29*s29_h1+pow28*s28_h1+pow27*s27_h1+pow26*s26_h1+pow25*s25_h1+pow24*s24_h1+pow23*s23_h1+pow22*s22_h1+pow21*s2
1_h1+pow20*s20_h1

+pow19*s19_h1+pow18*s18_h1+pow17*s17_h1+pow16*s16_h1+pow15*s15_h1+pow14*s14_h1+pow13*s13_h1+pow12*s12_h1+pow11*s1
1_h1+pow10*s10_h1

+pow09*s9_h1+pow08*s8_h1+pow07*s7_h1+pow06*s6_h1+pow05*s5_h1+pow04*s4_h1+pow03*s3_h1+pow02*s2_h1+pow01*s1_h1+pow0
0*s0_h1
      +pow08*cut0_h1+pow16*cut1_h1+pow24*cut2_h1;
    sum2 = pow33*s33_h2+pow32*s32_h2+pow31*s31_h2+pow30*s30_h2

+pow29*s29_h2+pow28*s28_h2+pow27*s27_h2+pow26*s26_h2+pow25*s25_h2+pow24*s24_h2+pow23*s23_h2+pow22*s22_h2+pow21*s2
1_h2+pow20*s20_h2

+pow19*s19_h2+pow18*s18_h2+pow17*s17_h2+pow16*s16_h2+pow15*s15_h2+pow14*s14_h2+pow13*s13_h2+pow12*s12_h2+pow11*s1
1_h2+pow10*s10_h2

+pow09*s9_h2+pow08*s8_h2+pow07*s7_h2+pow06*s6_h2+pow05*s5_h2+pow04*s4_h2+pow03*s3_h2+pow02*s2_h2+pow01*s1_h2+pow0
0*s0_h2
      +pow08*cut0_h2+pow16*cut1_h2+pow24*cut2_h2;
    sum3 = pow33*s33_h3+pow32*s32_h3+pow31*s31_h3+pow30*s30_h3

+pow29*s29_h3+pow28*s28_h3+pow27*s27_h3+pow26*s26_h3+pow25*s25_h3+pow24*s24_h3+pow23*s23_h3+pow22*s22_h3+pow21*s2
1_h3+pow20*s20_h3

+pow19*s19_h3+pow18*s18_h3+pow17*s17_h3+pow16*s16_h3+pow15*s15_h3+pow14*s14_h3+pow13*s13_h3+pow12*s12_h3+pow11*s1
1_h3+pow10*s10_h3

+pow09*s9_h3+pow08*s8_h3+pow07*s7_h3+pow06*s6_h3+pow05*s5_h3+pow04*s4_h3+pow03*s3_h3+pow02*s2_h3+pow01*s1_h3+pow0
0*s0_h3
      +pow08*cut0_h3+pow16*cut1_h3+pow24*cut2_h3;
    sum4 = pow33*s33_h4+pow32*s32_h4+pow31*s31_h4+pow30*s30_h4

+pow29*s29_h4+pow28*s28_h4+pow27*s27_h4+pow26*s26_h4+pow25*s25_h4+pow24*s24_h4+pow23*s23_h4+pow22*s22_h4+pow21*s2
1_h4+pow20*s20_h4

+pow19*s19_h4+pow18*s18_h4+pow17*s17_h4+pow16*s16_h4+pow15*s15_h4+pow14*s14_h4+pow13*s13_h4+pow12*s12_h4+pow11*s1
1_h4+pow10*s10_h4

+pow09*s9_h4+pow08*s8_h4+pow07*s7_h4+pow06*s6_h4+pow05*s5_h4+pow04*s4_h4+pow03*s3_h4+pow02*s2_h4+pow01*s1_h4+pow0
0*s0_h4
      +pow08*cut0_h4+pow16*cut1_h4+pow24*cut2_h4;
    sum5 = pow33*s33_h5+pow32*s32_h5+pow31*s31_h5+pow30*s30_h5

+pow29*s29_h5+pow28*s28_h5+pow27*s27_h5+pow26*s26_h5+pow25*s25_h5+pow24*s24_h5+pow23*s23_h5+pow22*s22_h5+pow21*s2
1_h5+pow20*s20_h5
```

```
+pow19*s19_h5+pow18*s18_h5+pow17*s17_h5+pow16*s16_h5+pow15*s15_h5+pow14*s14_h5+pow13*s13_h5+pow12*s12_h5+pow11*s1
1_h5+pow10*s10_h5

+pow09*s9_h5+pow08*s8_h5+pow07*s7_h5+pow06*s6_h5+pow05*s5_h5+pow04*s4_h5+pow03*s3_h5+pow02*s2_h5+pow01*s1_h5+pow0
0*s0_h5
      +pow08*cut0_h5+pow16*cut1_h5+pow24*cut2_h5;
    sum6 = pow33*s33_h6+pow32*s32_h6+pow31*s31_h6+pow30*s30_h6

+pow29*s29_h6+pow28*s28_h6+pow27*s27_h6+pow26*s26_h6+pow25*s25_h6+pow24*s24_h6+pow23*s23_h6+pow22*s22_h6+pow21*s2
1_h6+pow20*s20_h6

+pow19*s19_h6+pow18*s18_h6+pow17*s17_h6+pow16*s16_h6+pow15*s15_h6+pow14*s14_h6+pow13*s13_h6+pow12*s12_h6+pow11*s1
1_h6+pow10*s10_h6

+pow09*s9_h6+pow08*s8_h6+pow07*s7_h6+pow06*s6_h6+pow05*s5_h6+pow04*s4_h6+pow03*s3_h6+pow02*s2_h6+pow01*s1_h6+pow0
0*s0_h6
      +pow08*cut0_h6+pow16*cut1_h6+pow24*cut2_h6;
    sum7 = pow33*s33_h7+pow32*s32_h7+pow31*s31_h7+pow30*s30_h7

+pow29*s29_h7+pow28*s28_h7+pow27*s27_h7+pow26*s26_h7+pow25*s25_h7+pow24*s24_h7+pow23*s23_h7+pow22*s22_h7+pow21*s2
1_h7+pow20*s20_h7

+pow19*s19_h7+pow18*s18_h7+pow17*s17_h7+pow16*s16_h7+pow15*s15_h7+pow14*s14_h7+pow13*s13_h7+pow12*s12_h7+pow11*s1
1_h7+pow10*s10_h7

+pow09*s9_h7+pow08*s8_h7+pow07*s7_h7+pow06*s6_h7+pow05*s5_h7+pow04*s4_h7+pow03*s3_h7+pow02*s2_h7+pow01*s1_h7+pow0
0*s0_h7
      +pow08*cut0_h7+pow16*cut1_h7+pow24*cut2_h7;
    sum8 = pow33*s33_h8+pow32*s32_h8+pow31*s31_h8+pow30*s30_h8

+pow29*s29_h8+pow28*s28_h8+pow27*s27_h8+pow26*s26_h8+pow25*s25_h8+pow24*s24_h8+pow23*s23_h8+pow22*s22_h8+pow21*s2
1_h8+pow20*s20_h8

+pow19*s19_h8+pow18*s18_h8+pow17*s17_h8+pow16*s16_h8+pow15*s15_h8+pow14*s14_h8+pow13*s13_h8+pow12*s12_h8+pow11*s1
1_h8+pow10*s10_h8

+pow09*s9_h8+pow08*s8_h8+pow07*s7_h8+pow06*s6_h8+pow05*s5_h8+pow04*s4_h8+pow03*s3_h8+pow02*s2_h8+pow01*s1_h8+pow0
0*s0_h8
      +pow08*cut0_h8+pow16*cut1_h8+pow24*cut2_h8;
    sum9 = pow33*s33_h9+pow32*s32_h9+pow31*s31_h9+pow30*s30_h9

+pow29*s29_h9+pow28*s28_h9+pow27*s27_h9+pow26*s26_h9+pow25*s25_h9+pow24*s24_h9+pow23*s23_h9+pow22*s22_h9+pow21*s2
1_h9+pow20*s20_h9

+pow19*s19_h9+pow18*s18_h9+pow17*s17_h9+pow16*s16_h9+pow15*s15_h9+pow14*s14_h9+pow13*s13_h9+pow12*s12_h9+pow11*s1
1_h9+pow10*s10_h9

+pow09*s9_h9+pow08*s8_h9+pow07*s7_h9+pow06*s6_h9+pow05*s5_h9+pow04*s4_h9+pow03*s3_h9+pow02*s2_h9+pow01*s1_h9+pow0
0*s0_h9
      +pow08*cut0_h9+pow16*cut1_h9+pow24*cut2_h9;
    sum10 = pow33*s33_h10+pow32*s32_h10+pow31*s31_h10+pow30*s30_h10

+pow29*s29_h10+pow28*s28_h10+pow27*s27_h10+pow26*s26_h10+pow25*s25_h10+pow24*s24_h10+pow23*s23_h10+pow22*s22_h10+
pow21*s21_h10+pow20*s20_h10

+pow19*s19_h10+pow18*s18_h10+pow17*s17_h10+pow16*s16_h10+pow15*s15_h10+pow14*s14_h10+pow13*s13_h10+pow12*s12_h10+
pow11*s11_h10+pow10*s10_h10

+pow09*s9_h10+pow08*s8_h10+pow07*s7_h10+pow06*s6_h10+pow05*s5_h10+pow04*s4_h10+pow03*s3_h10+pow02*s2_h10+pow01*s1
_h10+pow00*s0_h10
      +pow08*cut0_h10+pow16*cut1_h10+pow24*cut2_h10;
    sum11 = pow33*s33_h11+pow32*s32_h11+pow31*s31_h11+pow30*s30_h11

+pow29*s29_h11+pow28*s28_h11+pow27*s27_h11+pow26*s26_h11+pow25*s25_h11+pow24*s24_h11+pow23*s23_h11+pow22*s22_h11+
pow21*s21_h11+pow20*s20_h11

+pow19*s19_h11+pow18*s18_h11+pow17*s17_h11+pow16*s16_h11+pow15*s15_h11+pow14*s14_h11+pow13*s13_h11+pow12*s12_h11+
pow11*s11_h11+pow10*s10_h11

+pow09*s9_h11+pow08*s8_h11+pow07*s7_h11+pow06*s6_h11+pow05*s5_h11+pow04*s4_h11+pow03*s3_h11+pow02*s2_h11+pow01*s1
_h11+pow00*s0_h11
      +pow08*cut0_h11+pow16*cut1_h11+pow24*cut2_h11;
    sum12 = pow33*s33_h12+pow32*s32_h12+pow31*s31_h12+pow30*s30_h12

+pow29*s29_h12+pow28*s28_h12+pow27*s27_h12+pow26*s26_h12+pow25*s25_h12+pow24*s24_h12+pow23*s23_h12+pow22*s22_h12+
pow21*s21_h12+pow20*s20_h12

+pow19*s19_h12+pow18*s18_h12+pow17*s17_h12+pow16*s16_h12+pow15*s15_h12+pow14*s14_h12+pow13*s13_h12+pow12*s12_h12+
pow11*s11_h12+pow10*s10_h12

+pow09*s9_h12+pow08*s8_h12+pow07*s7_h12+pow06*s6_h12+pow05*s5_h12+pow04*s4_h12+pow03*s3_h12+pow02*s2_h12+pow01*s1
_h12+pow00*s0_h12
      +pow08*cut0_h12+pow16*cut1_h12+pow24*cut2_h12;
    sum13 = pow33*s33_h13+pow32*s32_h13+pow31*s31_h13+pow30*s30_h13

+pow29*s29_h13+pow28*s28_h13+pow27*s27_h13+pow26*s26_h13+pow25*s25_h13+pow24*s24_h13+pow23*s23_h13+pow22*s22_h13+
pow21*s21_h13+pow20*s20_h13

+pow19*s19_h13+pow18*s18_h13+pow17*s17_h13+pow16*s16_h13+pow15*s15_h13+pow14*s14_h13+pow13*s13_h13+pow12*s12_h13+
pow11*s11_h13+pow10*s10_h13

+pow09*s9_h13+pow08*s8_h13+pow07*s7_h13+pow06*s6_h13+pow05*s5_h13+pow04*s4_h13+pow03*s3_h13+pow02*s2_h13+pow01*s1
_h13+pow00*s0_h13
      +pow08*cut0_h13+pow16*cut1_h13+pow24*cut2_h13;
    sum14 = pow33*s33_h14+pow32*s32_h14+pow31*s31_h14+pow30*s30_h14

+pow29*s29_h14+pow28*s28_h14+pow27*s27_h14+pow26*s26_h14+pow25*s25_h14+pow24*s24_h14+pow23*s23_h14+pow22*s22_h14+
pow21*s21_h14+pow20*s20_h14
```

```
+pow19*s19_h14+pow18*s18_h14+pow17*s17_h14+pow16*s16_h14+pow15*s15_h14+pow14*s14_h14+pow13*s13_h14+pow12*s12_h14+
pow11*s11_h14+pow10*s10_h14

+pow09*s9_h14+pow08*s8_h14+pow07*s7_h14+pow06*s6_h14+pow05*s5_h14+pow04*s4_h14+pow03*s3_h14+pow02*s2_h14+pow01*s1
_h14+pow00*s0_h14
      +pow08*cut0_h14+pow16*cut1_h14+pow24*cut2_h14;
    sum15 = pow33*s33_h15+pow32*s32_h15+pow31*s31_h15+pow30*s30_h15

+pow29*s29_h15+pow28*s28_h15+pow27*s27_h15+pow26*s26_h15+pow25*s25_h15+pow24*s24_h15+pow23*s23_h15+pow22*s22_h15+
pow21*s21_h15+pow20*s20_h15

+pow19*s19_h15+pow18*s18_h15+pow17*s17_h15+pow16*s16_h15+pow15*s15_h15+pow14*s14_h15+pow13*s13_h15+pow12*s12_h15+
pow11*s11_h15+pow10*s10_h15

+pow09*s9_h15+pow08*s8_h15+pow07*s7_h15+pow06*s6_h15+pow05*s5_h15+pow04*s4_h15+pow03*s3_h15+pow02*s2_h15+pow01*s1
_h15+pow00*s0_h15
      +pow00*ext0_h15+pow01*ext1_h15+pow01*ext2_h15;

    sout=pow00*sum0+pow02*sum1+pow04*sum2+pow06*sum3
        +pow08*sum4+pow10*sum5+pow12*sum6+pow14*sum7
        +pow16*sum8+pow18*sum9+pow20*sum10+pow22*sum11
        +pow24*sum12+pow26*sum13+pow28*sum14+pow30*sum15;

    sout=sout%pow64;

    if(sout>=pow63){
      sout=sout-pow64;
    }

    printf("   ain: %ld  bin: %ld  sout: %ld    expected_sout: %ld",ain,bin,sout,expected_sout);
    if(sout==expected_sout){printf("   allPASS\n");}
    else{printf("   allFAIL\n");}

  }
}
```

REFERENCES

1. L.Dadda,”Some schemes for parallel multipliers”,Alta Frequenza,Volume 34, pp. 346-356.

2. Whitney J.Townsend, Earl E. Swartzlander Jr.,and Jacob A. Abraham,”A comparison of Dadda and Wallace multiplier delays”,Advanced Signal Processing Algorithms,Architectures,and Implementations,Volume 5205, pp. 552-560,Austin,2003.

3. Hiro Mori,“A comparison of 32bx32b Dadda multiplier and “pre-sum before Dadda tree” multiplier.” github.com/bqe10133/multiplier_generator.git/20221018_presum_8bcut_12.pdf